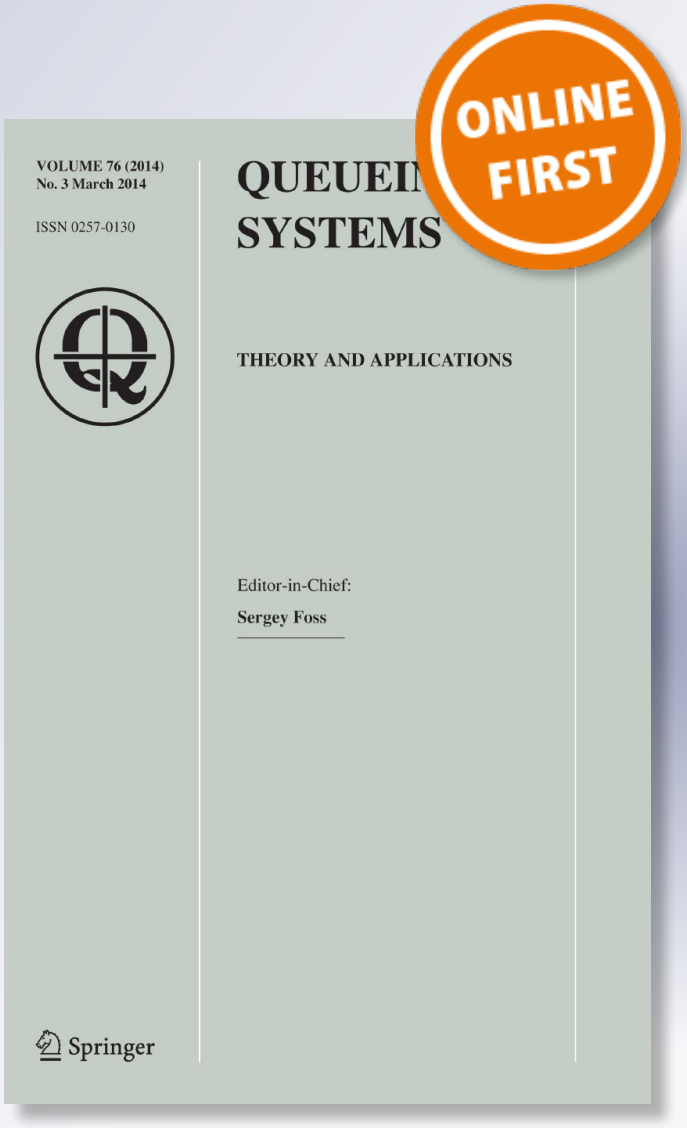
Perfect sampling of GI/GI/c *queues*

Jose Blanchet, Jing Dong & Yanan Pei

Queueing Systems
Theory and Applications

ISSN 0257-0130

Queueing Syst
DOI 10.1007/s11134-018-9573-2



Springer

Queueing Syst
https://doi.org/10.1007/s11134-018-9573-2

Perfect sampling of *GI/GI/c* queues

Jose Blanchet[1] · **Jing Dong**[2] · **Yanan Pei**[3]

Received: 7 October 2015 / Revised: 12 February 2018

Abstract We introduce the first class of perfect sampling algorithms for the steady-state distribution of multi-server queues with general interarrival time and service time distributions. Our algorithm is built on the classical dominated coupling from the past protocol. In particular, we use a coupled multi-server vacation system as the upper bound process and develop an algorithm to simulate the vacation system backward in time from stationarity at time zero. The algorithm has finite expected termination time with mild moment assumptions on the interarrival time and service time distributions.

Keywords Perfect sampling · FCFS multi-server queue · Dominated coupling from the past · Random walks

Mathematics Subject Classification 60K25

Support from NSF through the Grants CMMI-1538217, DMS-1320550 and DMS-1720433 is gratefully acknowledged.

✉ Jose Blanchet
jose.blanchet@stanford.edu

Jing Dong
jing.dong@gsb.columbia.edu

Yanan Pei
yp2342@columbia.edu

1 Management Science and Engineering, Stanford University, 475 Via Ortega, Suite 310, Stanford, CA 94305, USA

2 Graduate School of Business, Columbia University, 3022 Broadway, New York, NY 10027, USA

3 Department of IEOR, Columbia University, 500 West 120th Street, New York, NY 10027, USA

Published online: 13 March 2018

1 Introduction

In this paper, we present the first class of perfect sampling algorithms for the steady-state distribution of multi-server queues with general interarrival time and service time distributions. Our algorithm has finite expected running time under the assumption that the interarrival times and service times have finite $2+\epsilon$ moments for some $\epsilon > 0$.

The goal of perfect sampling is to sample without any bias from the steady-state distribution of a given ergodic process. The most popular perfect sampling protocol, known as Coupling From The Past (CFTP), was introduced by Propp and Wilson in the seminal paper [23]; see also [2] for another important early reference on perfect simulation. Foss and Tweedie [15] proved that CFTP can be applied if and only if the underlying process is uniformly ergodic, which is not a property applicable to multi-server queues. So, we use a variation of the CFTP protocol called Dominated CFTP (DCFTP) introduced by Kendall in [19] and later extended in [20,21].

A typical implementation of DCFTP requires at least four ingredients:

(a) a stationary upper bound process for the target process,
(b) a stationary lower bound process for the target process,
(c) the ability to simulate (a) and (b) backward in time (i.e., from time 0 to $-t$, for any $t > 0$),
(d) a finite time $-T < 0$ at which the state of the target process is determined (typically by having the upper and lower bound processes coalesce), and the ability to reconstruct the target process from $-T$ up to time 0 coupled with the two bounding processes.

The time $-T$ is called the coalescence time, and it is desirable to have $E[T] < \infty$. The ingredients are typically combined as follows. One simulates (a) and (b) backward in time (by applying (c)) until the processes meet. The target process is sandwiched between (a) and (b). Therefore, if we can find a time $-T < 0$ when processes (a) and (b) coincide, the state of the target process is known at $-T$ as well. Then, applying (d), we reconstruct the target process from $-T$ up to time 0. The algorithm outputs the state of the target process at time 0.

It is quite intuitive that the output of the above construction is stationary. Specifically, assume that the sample path of the target process coupled with (a) and (b) is given from $(-\infty, 0]$. Then, we can think of the simulation procedure in (c) as simply observing or unveiling the paths of (a) and (b) during $[-t, 0]$. When we find a time $-T < 0$ at which the paths of (a) and (b) take the same value, because of the sandwiching property, the target process must share this common value at $-T$. Starting from that point, property (d) simply unveils the path of the target process. Since this path has been coming from the infinite distant past (we simply observed it from time $-T$), the output is stationary at time 0. Notice that while $-T$ is a random time, the output is the state of the target process at the fixed time 0.

One can often improve the performance of a DCFTP protocol if the underlying target process is monotone [20], as in the multi-server queue setting. A process is monotone if there exists a certain partial order, $\preceq$, such that if w and w' are initial states where $w \preceq w'$, and one uses common random numbers to simulate two paths, one starting from w and the other from w', then the order is preserved when comparing

the states of these two paths at any point in time. Thus, instead of using the bounds (a) and (b) directly to detect coalescence, one could apply monotonicity to detect coalescence as follows: At any time $-t < 0$, one can start two paths of the target process, one from the state w' obtained from the upper bound (a) observed at time $-t$, and the other from the state $w \preceq w'$ obtained from the lower bound (b) observed at time $-t$. Then, we run these two paths using common random numbers, which are consistent with the backward simulation of (a) and (b), in reverse order according to the dynamics of the target process, and check whether these two paths meet before time zero. If they do, the coalescence occurs at such a meeting time. We also notice that because we are using common random numbers and system dynamics, these two paths will merge into a single path from the coalescence time forward, and the state at time zero will be the desired stationary draw. If coalescence does not occur, then one can simply let $t \leftarrow 2t$ and repeat the above procedure. For this iterative search procedure, we must show that the search terminates in finite time.

While the DCFTP protocol is relatively easy to understand, its application is not straightforward. In most applications, the most difficult part has to do with element (c). Then, there is an issue of finding good bounding processes (elements (a) and (b)), in the sense of having short coalescence times—which we interpret as making sure that $E[T] < \infty$. There has been a substantial amount of research that develops generic algorithms for Markov chains (see, for example, [10] and [8]). These methods rely on having access to the transition kernels, which are difficult to obtain in our case. Perfect simulation for queueing systems has also received a significant amount of attention in recent years, though most perfect simulation algorithms for queues impose Poisson assumptions on the arrival process. Sigman [25,26] applied the DCFTP and regenerative idea to develop perfect sampling algorithms for stable *M/G/c* queues. The algorithm in [25] requires the system to be super-stable (i.e., the system can be dominated by a stable *M/G/*1 queue). The algorithm in [26] works under natural stability conditions, but it has infinite expected termination time. A recent work by Connor and Kendall [9] extends Sigman's algorithm [26] to sample stationary *M/G/c* queues, and the algorithm has finite expected termination time, but it still requires the arrivals to be Poisson. The main reason for the Poisson arrival assumption is that under this assumption, one can find dominating processes which are quasi-reversible (see Chapter 3 of [18]) and therefore can be simulated backward in time using standard Markov chain constructions (element (c)).

In general, constructing elements (a) and (b), (a) in particular, as (b) can often be taken as the trivial lower bound, $\mathbf{0}$, in the multi-server queue setting requires proving sample path (almost sure) dominance under different service/routing disciplines. The sample path method has been widely used in the control of queues [22]. Comparison of multi-server queues, under the almost sure dominance or the stochastic dominance, has been studied in the literature (see, for example, [12,13,27] and references therein).

For general renewal arrival process, our work is close in the spirit to [4,11] and [6], but the model treated is fundamentally different. Thus, it requires some new developments. We also use a different coupling construction than that introduced in [26] and refined in [9]. In particular, we take advantage of a vacation system which allows us to transform the problem into simulating the running infinite horizon maximums (from time t to infinity) of renewal processes, compensated with negative drifts so

that the infinite horizon maximums are well defined. Finally, we note that a significant advantage of our method, in contrast to [26], is that we do not need to wait until the upper bound system empties to achieve coalescence. Due to the monotonicity of our process, we can apply the iterative method introduced above. This is important in many-server queues in heavy traffic for which it would take an exponential amount of time (in the arrival rate), or sometimes be impossible, to observe an empty system. We demonstrate the performance of our procedure for different many-server heavy traffic regimes using simulation experiments in Sect. 5.

The rest of the paper is organized as follows: In Sect. 2, we describe our simulation strategy, involving elements (a)–(d), and we conclude the section with the statement of a result which summarizes our main contribution (Theorem 1). Subsequent sections (Sects. 3 and 4) provide more details of our simulation strategy. In Sect. 5, we conduct some numerical experiments (an online companion of this paper includes a MATLAB implementation of the algorithm). Section 6 contains the proofs of some technical results. Lastly, we provide a list of selected notation in the Appendix.

2 Simulation strategy and main result

Our target process is the stationary process generated by a multi-server queue with independent and identically distributed (iid) interarrival times and iid service times which are independent of the arrivals. There are $c \geq 1$ identical servers, each can serve at most one customer at a time. Customers are served on a first-come-first-served (FCFS) basis. Let $G(\cdot)$ and $\bar{G}(\cdot) = 1 - G(\cdot)$ (resp. $F(\cdot)$ and $\bar{F}(\cdot) = 1 - F(\cdot)$) denote the cumulative distribution function, CDF, and the tail CDF of the interarrival times (resp. service times). We shall use A to denote a random variable with CDF G, and V to denote a random variable with CDF F.

Assumption 1 (**A1**) Both A and V are strictly positive with probability one, and there exists $\epsilon > 0$ such that

$$E[A^{2+\epsilon}] < \infty, \quad E[V^{2+\epsilon}] < \infty.$$

The previous assumption will allow us to conclude that the coalescence time of our algorithm has finite expectation. The algorithm will terminate with probability one if $E[A^{1+\epsilon}] + E[V^{1+\epsilon}] < \infty$.

We assume that $G(\cdot)$ and $F(\cdot)$ are known so that the required parameters in our algorithmic development can be obtained. We write $\lambda = (\int_0^\infty \bar{G}(t)\mathrm{d}t)^{-1} = 1/E[A]$ as the arrival rate, and $\mu = (\int_0^\infty \bar{F}(t)\mathrm{d}t)^{-1} = 1/E[V]$ as the service rate. In order to ensure the existence of the stationary distribution of the system, we require the following stability condition: $\lambda/(c\mu) < 1$.

2.1 Elements of the simulation strategy: upper bound and coupling

We refer to the upper bound process as the *vacation system*, the construction that we use is based on that given in [16]. Let us first explain in words how the vacation system

operates. Customers arrive at the vacation system according to the renewal arrival process, and the system operates similarly to a *GI/GI/c* queue, except that every time a server (say server i^*) finishes an activity (i.e., a service or a vacation), if there is no customer waiting to be served in the queue, server i^* takes a vacation which has the same distribution as the service times. If there is at least one customer waiting, the first customer waiting in the queue starts to be served by server i^*.

Using a suitable coupling, the work of [16] shows that the total number of jobs in the vacation system is an upper bound of the total number of jobs in the corresponding multi-server queue. In this paper, we establish bounds for other system-related processes, such as the Kiefer–Wolfowitz vectors, which are of independent interest.

We next provide more details about the vacation system. We introduce $(c + 1)$ time-stationary renewal processes, which are used to describe the vacation system.

Let

$$\mathcal{T}^0 := \left\{ T_n^0 : n \in \mathbb{Z}\backslash\{0\} \right\}$$

be a time-stationary renewal point process with $T_n^0 > 0$ and $T_{-n}^0 < 0$, $n \geq 1$ (the T_n^0 are sorted in a non-decreasing order in n). For $n \geq 1$, T_n^0 represents the arrival time of the n-th customer into the system after time zero, and T_{-n}^0 is the arrival time of the n-th customer, counting backward in time, from time zero. We also define

$$T_n^{0,+} := \inf \left\{ T_m^0 : T_m^0 > T_n^0 \right\},$$

that is, the arrival time of the next customer after T_n^0. If $n \geq 1$ or $n \leq -2$, $T_n^{0,+} = T_{n+1}^0$. However, $T_{-1}^{0,+} = T_1^0$. Similarly, we write

$$T_n^{0,-} := \sup \left\{ T_m^0 : T_m^0 < T_n^0 \right\},$$

i.e., the arrival time of the previous customer before T_n^0. Define $A_n := T_n^{0,+} - T_n^0$ for all $n \in \mathbb{Z}\backslash\{0\}$. Note that A_n is the interarrival time between the customer arriving at time T_n^0 and the next customer. A_n has CDF $G(\cdot)$ for $n \geq 1$ and $n \leq -2$, but A_{-1} has a different distribution due to the inspection paradox. Figure 1a provides a pictorial illustration of the renewal process $\mathcal{T}^0$.

Similarly, for $i \in \{1, 2, \ldots, c\}$, we introduce iid time-stationary renewal point processes

$$\mathcal{T}^i := \left\{ T_n^i : n \in \mathbb{Z}\backslash\{0\} \right\}.$$

As before, we have that $T_n^i > 0$ and $T_{-n}^i < 0$ for $n \geq 1$ with the T_n^i sorted in a non-decreasing order. We also define $T_n^{i,+} := \inf\{T_m^i : T_m^i > T_n^i\}$ and $T_n^{i,-} := \sup\{T_m^i : T_m^i < T_n^i\}$. Then, we let $V_n^i := T_n^{i,+} - T_n^i$. We assume that V_n^i has CDF $F(\cdot)$ for $n \geq 1$ and $n \leq -2$. The V_n^i are *activities* (services and vacations), which are executed by the i-th server in the vacation system.

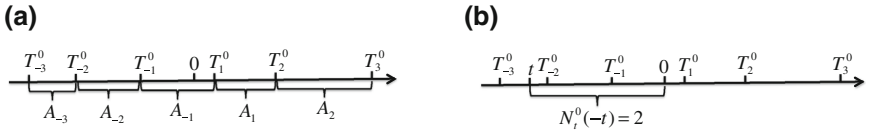


Fig. 1 Renewal processes. **a** Definition of A_i, **b** definition of $N^0_u(t)$

Next, we define, for each $i \in \{0, 1, \ldots, c\}$, and any $u \in (-\infty, \infty)$, a counting process

$$N^i_u(t) := \left| [u, u+t] \cap \mathcal{T}^i \right|,$$

for $t \geq 0$, where $|\ |$ denotes cardinality. Note that as $T^i_{-1} < 0 < T^i_1$ by stationarity, $N^i_0(0) = 0$. In particular, the quantity $N^0_u(t)$ is the number of customers who arrive during the time interval $[u, u+t]$ (see Fig. 1b). The quantity $N^i_u(t)$ is the number of activities initiated by server i during the time interval $[u, u+t]$ when $i \neq 0$. For simplicity of the notation, let us write $N^i(t) = N^i_0(t)$ if $t \geq 0$ and $N^i(t) = N^i_t(-t)$ if $t \leq 0$.

2.1.1 The upper bound process: vacation system

Let $Q_v(t)$ denote the number of people waiting in queue at time t in the stationary vacation system. We write $Q_v(t_-) := \lim_{s \uparrow t} Q_v(s)$ and $\mathrm{d}Q_v(t) := Q_v(t) - Q_v(t_-)$. Also, for any $t \geq 0$, $i \in \{0, \ldots, c\}$ and each $u \in (-\infty, \infty)$, define

$$N^i_u(t_-) := \lim_{h \downarrow 0} N^i_{u-h}(t),$$

and let $\mathrm{d}N^i_u(t) := N^i_u(t) - N^i_u(t_-)$ for all $t \geq 0$ (note that as $N^i_u(0_-) = 0$, $\mathrm{d}N^i_u(0)$ should equal $N^i_u(0)$). Similarly, for $t \leq 0$, $N^i(t_-) = N^i_t\left(|t|_-\right)$.

We also introduce $X_u(t) := N^0_u(t) - \sum_{i=1}^c N^i_u(t)$. For simplicity of the notation, we also write $X(t) = X_0(t)$ if $t \geq 0$, and $X(t) = X_t(-t)$ if $t \leq 0$. Then, the dynamics of $(Q_v(t) : t > 0)$ satisfy

$$\mathrm{d}Q_v(t) = \mathrm{d}X(t) + I(Q_v(t_-) = 0) \sum_{i=1}^{c} \mathrm{d}N^i(t), \tag{1}$$

given $Q_v(0)$. Note that here we are using the fact that arrivals do not occur at the same time as the start of activity times; this is because the processes $\mathcal{T}^i$ are independent time-stationary renewal processes in continuous time so that T^i_{-1} and T^i_1 have a density.

It follows from standard arguments of Skorokhod mapping [7] that, for $t \geq 0$,

$$Q_v(t) = Q_v(0) + X(t) - \inf_{0 \leq s \leq t} (X(s) + Q_v(0))^-,$$

where $(X(s) + Q_v(0))^- = \min(X(s) + Q_v(0), 0)$. Moreover, using Lyons construction, we have that, for $t \leq 0$,

$$Q_v(t) = \sup_{s \leq t} X(s) - X(t) \tag{2}$$

(see, for example, Proposition 1 of [3]). $(Q_v(t) : t \in (-\infty, \infty))$ is a well-defined process by virtue of the stability condition $\lambda/(\mu c) < 1$.

2.1.2 The coupling: extracting service times for each customer

The vacation system and the target process (the *GI/GI/c* queue) will be coupled by using the same arrival stream of customers, $\mathcal{T}^0$, and assuming that each customer brings his own service time. In particular, the evolution of the underlying *GI/GI/c* queue is described using a sequence of the form $\left(\left(T_n^0, V_n\right) : n \in \mathbb{Z}\backslash\{0\}\right)$, where V_n is the service time of the customer arriving at time T_n^0. In simulation, we start by simulating the upper bound process (vacation system). Thus, the V_n must be extracted from the evolution of $Q_v(\cdot)$ so that the same service times are matched to the common arrival stream both in the vacation system and in the target process.

In order to match the service times to each of the arriving customers in the vacation system, we define the following auxiliary processes: For every $i \in \{1, \ldots, c\}$, any $t > 0$, and any $u \in (-\infty, \infty)$, let $\sigma_u^i(t)$ denote the number of service initiations by server i during the time interval $[u, u + t]$. Observe that

$$\sigma_u^i(t) = \int_{[u,u+t]} I\left(Q_v(s_-) > 0\right) \mathrm{d}N_u^i(s - u).$$

That is, we count activity initiations at time $T_k^i \in [u, u+t]$ as service initiations if and only if $Q_v\left(T_{k-}^i\right) > 0$. Once again, here we use the fact that arrival times and activity initiation times do not occur simultaneously.

We now explain how to match service time for the customer arriving at T_n^0, $n \in \mathbb{Z}\backslash\{0\}$. First, such a customer occupies position $Q_v\left(T_n^0\right) \geq 1$ when he enters the queue. Let D_n^0 be the delay (or waiting time) inside the queue of the customer arriving at T_n^0. Then we have that

$$D_n^0 = \inf\left\{t \geq 0 : Q_v\left(T_n^0\right) = \sum_{i=1}^{c} \sigma_{T_n^0}^i(t)\right\},$$

and therefore,

$$V_n = \sum_{i=1}^{c} V_{N^i\left(T_n^0 + D_n^0\right)}^i \cdot \mathrm{d}N^i\left(T_n^0 + D_n^0\right). \tag{3}$$

Observe that the previous equation is valid, because for each $n \in \mathbb{Z}\backslash\{0\}$, there is a unique $i(n) \in \{1, \ldots, c\}$ for which $\mathrm{d}N^{i(n)}\left(T_n^0 + D_n^0\right) = 1$ and $\mathrm{d}N^j\left(T_n^0 + D_n^0\right) = 0$

if $j \neq i\,(n)$ (ties are not possible because of the time stationarity of the $\mathcal{T}^i$), so we obtain that (3) is equivalent to

$$V_n = V^{i(n)}_{N^{i(n)}(T^0_n + D^0_n)}.$$

We shall explain in Sect. 6.1 that $(V_n : n \in \mathbb{Z}\backslash\{0\})$ and $\left(T^0_n : n \in \mathbb{Z}\backslash\{0\}\right)$ are two independent sequences and the V_n are iid copies of V, i.e., the extraction procedure here does not create any bias.

2.2 Monotonicity properties and the stationary *GI/GI/c* queue

2.2.1 A family of GI/GI/c queues and the target GI/GI/c stationary system

We now describe the evolution of a family of standard *GI/GI/c* queues. Once we have the sequence $\left(\left(T^0_n, V_n\right) : n \in \mathbb{Z}\backslash\{0\}\right)$, we can proceed to construct a family of continuous-time Markov processes $(Z_u(t; z) : t \geq 0)$ for each $u \in (-\infty, \infty)$, given the initial condition $Z_u(0; z) = z$. We write $z = (q, r, e(u))$, and set

$$Z_u(t; z) := \left(Q_u(t; z), R_u(t; z), E_u(t; z)\right),$$

for $t \geq 0$, where $Q_u(t; z)$ is the number of people in the queue at time $u+t$ ($Q_u(0; z) = q$), $R_u(t; z)$ is the vector of ordered (ascending) remaining service times of the c servers at time $u + t$ ($R_u(0; z) = r$), and $E_u(t; z)$ is the time elapsed since the previous arrival at time $u + t$ ($E_u(0; z) = e(u)$).

We shall always use $E_u(0; z) = e(u) = u - \sup\{T^0_n : T^0_n \leq u\}$, and we shall select q and r appropriately based on the upper bound. The evolution of the process $(Z_u(s; z) : 0 < s \leq t)$ is obtained by feeding the traffic $\{\left(T^0_n, V_n\right) : u < T^0_n \leq u + s\}$ for $s \in (0, t]$ into a FCFS *GI/GI/c* queue with initial conditions given by z. Constructing $(Z_u(s; z) : 0 < s \leq t)$ using the traffic trace $\{\left(T^0_n, V_n\right) : u < T^0_n \leq u + s\}$ for $s \in (0, t]$ is standard (see, for example, Chapter 3 of [24]).

One can further describe the evolution of the underlying *GI/GI/c* queue at arrival epochs, using the Kiefer–Wolfowitz vector [1]. In particular, for every non-negative vector $w \in \mathbb{R}^c$ such that $w^{(i)} \leq w^{(i+1)}$ (where $w^{(i)}$ is the i-th entry of w) for $1 \leq i \leq c - 1$, and each $k \in \mathbb{Z}\backslash\{0\}$, the family of processes $\{W_k\left(T^0_n; w\right) : n \geq k, n \in \mathbb{Z}\backslash\{0\}\}$ satisfies

$$W_k\left(T^{0,+}_n; w\right) = \mathcal{S}\left(\left(W_k\left(T^0_n; w\right) + V_n \mathbf{e}_1 - A_n \mathbf{1}\right)^+\right), \tag{4}$$

with initial condition $W_k\left(T^0_k; w\right) = w$, where $\mathbf{e}_1 = (1, 0, \ldots, 0)^T \in \mathbb{R}^c$, $\mathbf{1} = (1, \ldots, 1)^T \in \mathbb{R}^c$, and $\mathcal{S}$ is a sorting operator which arranges the entries in a vector in ascending order. In simple words, $W_k\left(T^0_n; w\right)$ for $k \in \mathbb{Z}\backslash\{0\}$ describes the Kiefer–Wolfowitz vector as observed by the customer arriving at T^0_n, assuming that customer who arrived at T^0_k, $k \leq n$, experienced the Kiefer–Wolfowitz state w.

Recall that the first entry of $W_k\left(T^0_n; w\right)$, namely $W^{(1)}_k\left(T^0_n; w\right)$, is the waiting time of the customer arriving at T^0_n (given the initial condition w at T^0_k). More generally,

the i-th entry of $W_k\left(T_n^0; w\right)$, namely $W_k^{(i)}\left(T_n^0; w\right)$, is the virtual waiting time of the customer arriving at T_n^0 if he decides to enter service immediately after there are at least i servers free once he reaches the head of the line. In other words, one can also interpret $W_k\left(T_n^0; w\right)$ as the vector of remaining workloads (sorted in ascending order) that would be processed by each of the c servers at T_n^0, if there are no more arrivals after time T_n^0.

We are now ready to construct the stationary version of the *GI/GI/c* queue. Namely, for each $n \in \mathbb{Z}\backslash\{0\}$ and every $t \in (-\infty, \infty)$, we define $W(n)$ and $Z(t)$ via

$$\begin{aligned} W(n) &:= \lim_{k\to-\infty} W_k\left(T_n^0; 0\right), \\ Z(t) &:= (Q(t), R(t), E(t)) = \lim_{u\to-\infty} Z_u(t-u, z_-(u)), \end{aligned} \tag{5}$$

where $z_-(u) = (0, 0, e(u))$. We shall show in Proposition 1 that these limits are well defined.

2.2.2 The analogue of the Kiefer–Wolfowitz process for the upper bound system

In order to complete the coupling strategy, we also describe the evolution of the analogous Kiefer–Wolfowitz vector induced by the vacation system, which we denote by $(W_v(n) : n \in \mathbb{Z}\backslash\{0\})$, where v stands for vacation. As with the i-th entry of the Kiefer–Wolfowitz vector of a *GI/GI/c* queue, the i-th entry of $W_v(n)$, namely $W_v^{(i)}(n)$, is the virtual waiting time of the customer arriving at time T_n^0 if he decides to enter service immediately after there are at least i servers free once he reaches the head of the line (assuming that servers become idle once they see, after the completion of current activity, the customer in queue waiting in the head of the line).

To describe the Kiefer–Wolfowitz vector induced by the vacation system precisely, let $U^i(t)$ be the time until the next renewal after time t in $\mathcal{T}^i$, that is $U^i(t) = \inf\{T_n^i : T_n^i > t\} - t$. So, for example, $U^0\left(T_n^0\right) = A_n$ for $n \in \mathbb{Z}\backslash\{0\}$. Let $U(t) = \left(U^1(t), \ldots, U^c(t)\right)^T$.

We then have that

$$W_v(n) = D_n^0\mathbf{1} + \mathcal{S}\left(U\left(\left(T_n^0 + D_n^0\right)_-\right)\right). \tag{6}$$

In particular, note that $W_v^{(1)}(n) = D_n^0$, i.e., the delay the customer arriving at T_n^0 would experience. We next introduce a recursive way of constructing/defining the Kiefer–Wolfowitz vector induced by the vacation system. We define

$$\bar{W}_v(n) = W_v(n) + V_n\mathbf{e}_1 - A_n\mathbf{1},$$

and let $\bar{W}_v^{(i)}(n)$ be the i-th entry of $\bar{W}_v(n)$. Let $W_v(n+)$ denote the Kiefer–Wolfowitz vector seen by the customer arriving at $T_n^{0,+}$. From the definition of $W_v(n)$, we have

$$W_v(n+) = \mathcal{S}\left(\left(\bar{W}_v(n)\right)^+ + \Xi_n\right),$$

where

$$\Xi_n^{(i)} = I(\bar{W}_v^{(i)}(n) < 0) \cdot U^{j_i(n)}\left(\left(T_n^{0,+}\right)_-\right)$$

(i.e., $j_i(n)$ is the server whose remaining activity time immediately before T_n^0 is the i-th smallest in order).

So, (6) actually satisfies

$$W_v(n+) = \mathcal{S}\left((W_v(n) + V_n\mathbf{e}_1 - A_n\mathbf{1})^+ + \Xi_n\right), \tag{7}$$

where $\Xi_n = \left(\Xi_n^{(1)}, \ldots, \Xi_n^{(c)}\right)^T$.

2.2.3 Monotonicity properties

In this section, we will present several lemmas which contain useful monotonicity properties. The proofs of the lemmas are given in Sect. 6.2 in order to quickly arrive at the main point of this section, which is the construction of a stationary version of the *GI/GI/c* queue.

First, we recall that the Kiefer–Wolfowitz vector of a *GI/GI/c* queue is monotone in the initial condition (8) and invoke a property (9) which will allow us to construct a stationary version of the Kiefer–Wolfowitz vector of our underlying *GI/GI/c* queue, using Lyons construction.

Lemma 1 *For $n \geq k$, $k, n \in \mathbb{Z}\backslash\{0\}$, $w^+ > w^-$,*

$$W_k\left(T_n^0; w^+\right) \geq W_k\left(T_n^0; w^-\right). \tag{8}$$

Moreover, if $k \leq k' \leq n$,

$$W_k\left(T_n^0; \mathbf{0}\right) \geq W_{k'}\left(T_n^0; \mathbf{0}\right). \tag{9}$$

The second result allows us to make precise how the vacation system dominates a suitable family of *GI/GI/c* systems, in terms of the underlying Kiefer–Wolfowitz vectors.

Lemma 2 *For $n \geq k$, $k, n \in \mathbb{Z}\backslash\{0\}$,*

$$W_v(n) \geq W_k\left(T_n^0; W_v(k)\right).$$

The next result shows that in terms of queue length processes, the vacation system also dominates a family of *GI/GI/c* queues, which we shall use to construct the upper bounds.

Lemma 3 *Let $q = Q_v(u)$, $r = \mathcal{S}(U(u))$, and $e = u - \sup\{T_n^0 : T_n^0 \leq u\}$, so that $z^+ = (q, r, e)$ and $z^- = (0, \mathbf{0}, e)$. Then, for $t \geq u$,*

$$Q_u(t-u;z^-) \leq Q_u(t-u;z^+) \leq Q_v(t).$$

Using Lemmas 1–3, we can establish the following result.

Proposition 1 *The limits defining $W(n)$ and $Z(t)$ in (5) exist almost surely. Moreover, we have*

$$W_k\left(T_n^0;\mathbf{0}\right) \leq W(n) \leq W_k\left(T_n^0;W_v(k)\right). \tag{10}$$

Proof Using Lemmas 1 and 2, we have that

$$W_v(n) \geq W_k\left(T_n^0;W_v(k)\right) \geq W_k\left(T_n^0;\mathbf{0}\right).$$

Then, by property (9) in Lemma 1, we conclude that the limit defining $W(n)$ exists almost surely and that

$$W(n) \leq W_v(n). \tag{11}$$

Similarly, using Lemma 3, we can obtain the existence of the limit $Q(t)$ and we have that $Q(t) \leq Q_v(t)$. Moreover, by convergence of the Kiefer–Wolfowitz vectors, we obtain the i-th entry of $R\left(T_n^0 + W^{(1)}(n)\right)$, namely

$$R^{(i)}\left(T_n^0 + W^{(1)}(n)\right) = \left(W^{(i)}(n) - W^{(1)}(n)\right)^+,$$

where $i \in \{1,\ldots,c\}$. Lastly, since the age process has been taken from the underlying renewal process $\mathcal{T}^0$, we have that $E(t) = t - \sup\{T_n^0 : T_n^0 \leq t\}$. The fact that the limits are stationary follows directly from the limiting procedure and is standard in Lyons-type constructions.

For (10), we use the identity $W(n) = W_k\left(T_n^0;W(k)\right)$, combined with Lemma 1, to obtain

$$W_k\left(T_n^0;\mathbf{0}\right) \leq W_k\left(T_n^0;W(k)\right) = W(n),$$

and then we apply Lemma 2, together with (11), to obtain

$$W(n) = W_k\left(T_n^0;W(k)\right) \leq W_k\left(T_n^0;W_v(k)\right).$$

□

2.3 Description of simulation strategy and main result

We now describe how the variation of DCFTP that we mentioned in the Introduction, using monotonicity of the multi-server queue, and elements (a)–(d), apply to our setting.

Define a fixed inspection sequence $\{\kappa_j : j \geq 1\}$ with $\kappa_j < \kappa_{j-1} < 0$, and define $\kappa_0 = 0$. We start from the first inspection time $T_{\kappa_1}^0$ $(j = 1)$. The upper bound is initialized using the Kiefer–Wolfowitz process associated with the vacation system at $T_{\kappa_j}^0$. The lower bound is initialized with a null vector $\mathbf{0}$. We run the two bounding

GI/GI/c queues forward in time using $\{(T_n^0, V_n) : \kappa_j \leq n \leq \kappa_{j-1}\}$. If the two processes meet before time zero, then we can "unveil" the state of the stationary *GI/GI/c* queue; otherwise, we go backward in time to the next inspection time $T^0_{\kappa_{j+1}}$ $(j \leftarrow j+1)$ and construct two new bounding *GI/GI/c* queues accordingly. We repeat the procedure until the coalescence is detected.

The strategy combines the following facts (which we shall discuss in the sequel).

- *Fact I* We can simulate $\sup_{s \geq t} X(-s)$ and $(N^i(-t) : t \geq 0)_{i=0}^c$ jointly for any given $t \geq 0$. This part, which corresponds to item (c) is executed by applying an algorithm from [6] designed to sample the infinite horizon running time maximum of a random walk with negative drift. We shall provide more details about this in Sect. 4.
- *Fact II* For all $k \leq -1$ and every $k \leq n \leq -1$, by Proposition 1, we have that

$$W_k\left(T_n^0; \mathbf{0}\right) \leq W(n) \leq W_k\left(T_n^0; W_v(k)\right).$$

  This portion exploits the upper bound (a) (i.e., $W_v(k)$) and the lower bound (b) (i.e., $\mathbf{0}$).
- *Fact III* We can detect that coalescence occurs at some time $T \in [T_k^0, 0]$ for some $k \leq -1$ by finding some $n \in \mathbb{Z}_-$, $n \geq k$, such that $T_n^0 + W_k^{(1)}\left(T_n^0; W_v(k)\right) \leq 0$ and

$$W_k\left(T_n^0; W_v(k)\right) = W_k\left(T_n^0; 0\right).$$

  This is precisely the coalescence detection strategy which uses monotonicity of the Kiefer–Wolfowitz vector and the coalescence time $T = T_n^0 + W_k^{(1)}\left(T_n^0; W_v(k)\right)$.
- *Fact IV* We can combine Facts I–III to conclude that

$$Z^0_{T_k}\left(\left|T_k^0\right|; Q\left(T_k^0\right), \mathcal{S}\left(U\left(T_k^0\right)\right), 0\right) = Z(0) \tag{12}$$

  is stationary. We also have that

$$W_k\left(T_1^0; 0\right) = W(1),$$

  which follows the stationary distribution of the Kiefer–Wolfowitz vector of a *GI/GI/c* queue.

The main result of this paper is stated in the following theorem.

Theorem 1 *Assume (A1) is in force, with* $\lambda/(c\mu) \in (0, 1)$. *Then, Facts I–IV hold true. We can detect coalescence at a time* $-T < 0$ *such that* $E(T) < \infty$.

The rest of the paper is dedicated to the proof of Theorem 1. We have verified a number of monotonicity properties in Sect. 2.2.3, which in particular allow us to conclude that the construction of $W(n)$ and $Z(t)$ is legitimate (i.e., the limits exist almost surely). The monotonicity properties also yield Fact II and pave the way to verify Fact III. Section 3 proves the finite expectation of the coalescence time. In Sect. 4, we provide more algorithmic details about our perfect sampling construction.

3 Coalescence detection in finite time

In this section, we give more details about the coalescence detection scheme. The next result corresponds to Fact III and Fact IV.

Proposition 2 *Suppose that* $w^+ = W_v(k)$ *for some* $k \leq -1$, *and* $w^- = \mathbf{0}$. *If* $W_k\left(T_n^0; w^+\right) = W_k\left(T_n^0; w^-\right)$ *for some* $k \leq n \leq -1$, *then* $W_k\left(T_m^0; w^+\right) = W(m) = W_k\left(T_m^0; w^-\right)$ *for all* $m \geq n$. *Moreover, for all* $t \geq T_n^0 + W_k^{(1)}\left(T_n^0; w^+\right)$,

$$Z_{T_k^0}\left(t - T_k^0; \left(Q_v\left(T_k^0\right), \mathcal{S}\left(U\left(T_k^0\right)\right), 0\right)\right) = Z_{T_k^0}\left(t - T_k^0; (0, \mathbf{0}, 0)\right) = Z(t). \tag{13}$$

Proof The fact that

$$W_k\left(T_m^0; w^+\right) = W(m) = W_k\left(T_m^0; w^-\right)$$

for $m \geq n$ follows immediately from the recursion defining the Kiefer–Wolfowitz vector. Now, to show the first equality in (13), it suffices to consider $t = T_n^0 + W_k^{(1)}\left(T_n^0; w^+\right)$, since from $t \geq T_n^0$ the input is exactly the same and everyone coming after T_n^0 will depart the queue and enter service after time $T_n^0 + W_k^{(1)}\left(T_n^0; w^+\right)$. The arrival processes (i.e., $E_u(\cdot)$) clearly agree, so we just need to verify that the queue lengths and the residual service times agree. First, note that

$$\begin{aligned}
&R_{T_k^0}\left(T_n^0 + W_k^{(1)}\left(T_n^0; w^+\right) - T_k^0; \left(Q_v\left(T_k^0\right), \mathcal{S}\left(U\left(T_k^0\right)\right), 0\right)\right) \\
&\quad = W_k\left(T_n^0; w^+\right) - W_k^{(1)}\left(T_n^0; w^+\right) \cdot \mathbf{1} \\
&\quad = W_k\left(T_n^0; w^-\right) - W_k^{(1)}\left(T_n^0; w^-\right) \cdot \mathbf{1} \\
&\quad = R_{T_k^0}\left(T_n^0 + W_k^{(1)}\left(T_n^0; w^-\right) - T_k^0; (0, \mathbf{0}, 0)\right).
\end{aligned} \tag{14}$$

So, the residual service times of both upper and lower bound processes agree. The agreement of the queue lengths follows from Lemma 3. Finally, the second equality in (13) follows from Proposition 1. □

Next, we analyze properties of the coalescence time. Define

$$\begin{aligned}
T_- = \sup\Bigg\{ T_k^0 \leq 0 : \inf_{T_k^0 \leq t \leq 0} \Big\| & Z_{T_k^0}\left(t - T_k^0; \left(Q_v\left(T_k^0\right), \mathcal{S}\left(U\left(T_k^0\right)\right), 0\right)\right) \\
& - Z_{T_k^0}\left(t - T_k^0; (0, \mathbf{0}, 0)\right)\Big\|_\infty = 0 \Bigg\}.
\end{aligned}$$

Notice that if at time T_- we start an upper bound queue,

$$Z_{T_-}(\cdot; (Q_v(T_-), \mathcal{S}(U(T_-)), 0)),$$

and a lower bound queue, $Z_{T_-}(\cdot;(0,\mathbf{0},0))$, they will coalesce before time 0. Thus, if we simulate the system up to T_-, we will be able to detect a coalescence. We next establish that $E[|T_-|]<\infty$.

By stationarity, we have that $|T_-|$ is equal in distribution to

$$T=\inf\left\{T_k^0\geq 0:\inf_{0\leq t\leq T_k^0}\|Z_0(t;(Q_v(0),\mathcal{S}(U(0)),0))-Z_0(t;(0,\mathbf{0},0))\|_\infty=0\right\}.$$

Proposition 3 *If $E[V_n]<cE[A_n]$ for $n\geq 1$ and Assumption (A1) holds,*

$$E[T]<\infty.$$

Proof Define

$$\tau=\inf\left\{n\geq 1: W_1\left(T_n^0;W_v(1)\right)=W_1\left(T_n^0;0\right)\right\}.$$

By Wald's identity, $EA_n<\infty$, for any $n\geq 1$; it suffices to show that $E[\tau]<\infty$.

We start with an outline of the proof, which involves two main components. I) We first construct a sequence of events which lead to the occurrence of τ. The events that we construct put constraints on the interarrival times and service times so that we see a decreasing trend on the Kiefer–Wolfowitz vectors. When putting a number of these events together (consecutively), the waiting time of the upper bound system will drop to zero. We further impose the events for c more arrivals after the waiting time drops to zero. Notice that these c arrivals do not have to wait in both the upper bound and the lower bound systems. Thus, by the time of c-th such arrival, the two systems will have the same set of customers with the same remaining service times. II) Based on events constructed in I, we then split the process $\{W_1(T_n^0;W_v(1)):n\geq 1\}$ into cycles where: IIa) the probability that the desired event, which leads to coalescence, happens during each cycle is bounded from below by a positive constant, and IIb) the expected cycle length is bounded from above by a constant. IIa allows us to bound the number of cycles we need to check before finding τ by a geometric random variable. Then, we apply Wald's identity using IIb to establish an upper bound for $E[\tau]$.

We next provide more details of the proof, which are divided into part I and II as outlined above.

Part I We first construct the sequence of events, $\{\Omega_k:k\geq 2\}$, which enjoys the property that if Ω_k happens, the two bounding systems will have coalesced by time of the $(k+\lceil cK/\epsilon\rceil-1)$-th arrival.

As $E[V_n]<cE[A_n]$, for $n\geq 2$, we can find $m,\epsilon>0$ such that for every $n\geq 2$, the event $H_n=\{V_n<cm-\epsilon,A_n>m\}$ is nontrivial in the sense that $P(H_n)>\delta$ for some $\delta>0$. Now, pick $K>cm$ large enough, and define, for $k\geq 2$,

$$\Omega_k=\left\{W_1^{(c)}\left(T_k^0;W_v(1)\right)\leq K\right\}\bigcap_{n=k}^{k+\lceil cK/\epsilon\rceil-1}H_n.$$

To see the coalescence of the two bounding systems, let $\tilde{W}_k = (K, K, \ldots, K)^T$ be a c-dimensional vector with each element equal to K. We notice that, under Ω_k,

$$\tilde{W}_k \geq W_1\left(T_k^0; W_v(1)\right).$$

For $n \geq k$, define $\tilde{V}_n = cm - \epsilon$, $\tilde{A}_n = m$, and the (auxiliary) Kiefer–Wolfowitz sequence

$$\tilde{W}_{n+1} = \mathcal{S}\left(\left(\tilde{W}_n + \tilde{V}_n \mathbf{e}_1 - \tilde{A}_n \mathbf{1}\right)^+\right).$$

Then, Ω_k implies $V_n < \tilde{V}_n$ and $A_n > \tilde{A}_n$ for $n \geq k$, which in turn implies

$$W_1\left(T_n^0; W_v(1)\right) \leq \tilde{W}_n.$$

Moreover, under Ω_k, we have

$$\tilde{W}_n^{(1)} = 0 \text{ and } \tilde{W}_n^{(c)} < cm \text{ for } n = k + \lceil cK/\epsilon \rceil - c + 1, \ldots, k + \lceil cK/\epsilon \rceil.$$

Then, $W_1^{(1)}\left(T_n^0; W_v(1)\right) = 0$ and $W_1^{(c)}\left(T_n^0; W_v(1)\right) < cm$ for $n = k + \lceil cK/\epsilon \rceil - c + 1, \ldots, k + \lceil cK/\epsilon \rceil$. This indicates that under Ω_k, (1) all the arrivals between the $(k + \lceil cK/\epsilon \rceil - c + 1)$-th arrival and the $(k + \lceil cK/\epsilon \rceil)$-th arrival (included) enter service immediately upon arrival (have zero waiting time), and (2) the customers initially seen by the $(k + \lceil cK/\epsilon \rceil - c + 1)$-th arrival would have left the system by the time of the $(k + \lceil cK/\epsilon \rceil)$-th arrival. The same analysis holds assuming that we replace $W_1\left(T_k^0; W_v(1)\right)$ by $W_1\left(T_k^0; 0\right)$. Therefore, by the time of the $(k + \lceil cK/\epsilon \rceil - 1)$-th arrival, the two bounding systems would have exactly the same set of customers with exactly the same remaining service times, which is equal to their service times minus the time elapsed since their arrival times (since all of them start service immediately upon arrival). We also notice that since there is no customer waiting, the sorted remaining service time at $T^0_{k+\lceil cK/\epsilon \rceil - 1}$ coincides with the Kiefer–Wolfowitz vector $W_{k+\lceil cK/\epsilon \rceil - 1}$.

Part II We first introduce how to split the process into cycles, which are denoted as $\{(\tilde{\kappa}_i, \tilde{\kappa}_{i+1}), i \geq 1\}$. Let $\mathcal{U}_K := \{w : w^{(c)} \leq K\}$. We define

$$\tilde{\kappa}_1 := \inf\left\{n \geq 1 : W_1\left(T_n^0; W_v(1)\right) \in \mathcal{U}_K\right\},$$

and for $i \geq 2$, define

$$\tilde{\kappa}_i := \left\{n > \tilde{\kappa}_{i-1} + \lceil cK/\epsilon \rceil - 1 : W_1(T_n^0; W_v(1)) \in \mathcal{U}_K\right\}.$$

We denote $\Theta_i = \bigcap_{n=\tilde{\kappa}_i}^{\tilde{\kappa}_i + \lceil cK/\epsilon \rceil - 1} H_n$ for $i \geq 1$. We next show that the event Θ_i happens during the i-th cycle with positive probability. Since $P(H_n) > \delta$, $P(\Theta_i) \geq \delta^{\lceil cK/\epsilon \rceil} >$

0. Let N denote the first i for which Θ_i occurs. Then, N is stochastically bounded by a geometric random variable with probability of success $\delta^{\lceil cK/\epsilon \rceil}$. In particular, $E[N] \leq \delta^{-\lceil cK/\epsilon \rceil} < \infty$.

We next show that $E[\tilde{\kappa}_{i+1} - \tilde{\kappa}_i]$ is bounded using the standard Lyapunov argument. Under Assumption (A1) and $\lambda < c\mu$, $\{W_1\left(T_n^0; w\left(1\right)\right) : n \geq 1\}$ for any fixed initial condition $w(1)$ is a positive recurrent Harris chain [1]. Under Assumption (A1), we also have that $(Q_v(t) : t \in (-\infty, \infty))$ is a well-defined process with $E[Q_v(t)] < \infty$ (see the random-walk bound in (18)). Thus,

$$E\left[\sum_{i=1}^{c} W_v^{(i)}\left(1\right)\right] < \infty.$$

Consider the Lyapunov function $g(W) = W^{(c)}$, i.e., $g(W) \geq 0$ and $g(W) \to \infty$ as $||W|| \to \infty$. Then, for K large enough, as $\lambda < c\mu$, we can find $\delta \in (0, c/\lambda - 1/\mu)$ such that

$$E\left[g\left(W_1(T_{c+1}^0, w(1))\right)\right] \leq g(w(1)) - \delta \text{ for } w(1) \notin \mathcal{U}_K. \tag{15}$$

We also have

$$E\left[g\left(W_1(T_{c+1}^0, w(1))\right)\right] \leq K + c/\mu \text{ for } w(1) \in \mathcal{U}_K.$$

Then, by Theorem 2 in [14], $E[\tilde{\kappa}_1] < \infty$ and we can find a constant $M > 0$ such that $E[\tilde{\kappa}_i - \tilde{\kappa}_{i-1}] < M$ for $i \geq 2$. We comment that here we need to look c steps ahead to identify the downward drift in (15), Thus, we use a general version of the Lyapunov argument developed in [14].

Lastly, by Wald's identity, we have (setting $\tilde{\kappa}_0 = 0$) that

$$\begin{aligned} E[\tau] &\leq E[\tilde{\kappa}_N] + \lceil cK/\epsilon \rceil - 1 \\ &= E\sum_{i=1}^{N}(\tilde{\kappa}_i - \tilde{\kappa}_{i-1}) + \lceil cK/\epsilon \rceil - 1 \\ &\leq E[N] \times M + E[\tilde{\kappa}_1] + \lceil cK/\epsilon \rceil - 1 < \infty. \end{aligned}$$

□

Remark 1 Following the proof, we can also conclude that the number of "activities" (either vacations or services) to simulate in the vacation system, denoted by N_V, is also finite in expectation. Since coalescence is detected by the τ-th arrival, we only need to simulate the vacation system forward in time from time 0 until we are able to extract the first $Q_v(0) + \tau$ service time requirements to match the customers waiting in queue at time 0 and the arrivals from time 0 to coalescence time T.

For any $m' < \infty$ such that $\mathbb{E}[V \wedge m'] > 0$, we let $\bar{N}^i(t)$, $i = 1, \ldots, c$, denote the counting process corresponding to the i-th "truncated" vacation process with independent activity times capped by m', i.e., $V \wedge m'$. Following a standard argument as in the proof of Ward's identity in [1], a loose upper bound for $E[N_v]$ is given by

$$
\begin{aligned}
E[N_V] &\le \sum_{i=1}^{c} E\left[N^i(T)+1\right] + E[Q_v(0)+\tau] \\
&\le \sum_{i=1}^{c} E\left[\bar{N}^i(T)+1\right] + E[Q_v(0)] + E[\tau] \\
&\le c \cdot \frac{E[T]+m'}{E[V \wedge m']} + E[Q_v(0)] + E[\tau] < \infty.
\end{aligned}
$$

4 Simulation procedure

In this section, we first address the validity of Fact I, namely, that we can simulate the vacation system backward in time, jointly with $\left\{T_n^i : m \le n \le -1\right\}$ for $0 \le i \le c$, for any $m \in \mathbb{Z}_-$.

Let $G_e(\cdot) = \lambda \int_0^{\cdot} \bar{G}(x)\mathrm{d}x$ and $F_e(\cdot) = \mu \int_0^{\cdot} \bar{F}(x)\mathrm{d}x$ denote equilibrium CDFs of the interarrival time and service time distributions, respectively. We first notice that simulating the stationary arrival process $\left\{T_n^0 : n \le -1\right\}$ and stationary service/vacation completion process $\left\{T_n^i : n \le -1\right\}$ for each $1 \le i \le c$ is straightforward by the reversibility of $\mathcal{T}_n^i$ for $0 \le i \le c$. Specifically, we can simulate the renewal arrival process forward in time from time 0 with the first interarrival time following G_e and subsequent interarrival times following G. We then set $T_{-k}^0 = -T_k^0$ for all $k \ge 1$. Likewise, we can also simulate the service/vacation process of server i, for $i = 1, \ldots, c$, forward in time from time 0 with the first service/vacation initiation time following F_e and subsequent service/vacation time requirements distributed as F. Let T_k^i, $k \ge 1$, denote the k-th service/vacation initiation time of server i counting forward in time. Then, we set $T_{-k}^i = -T_k^i$.

Similarly, we have the equality in distribution, for all $t \ge 0$ (jointly),

$$
X(-t) \stackrel{d}{=} X(t);
$$

therefore, we have from (2) that the following equality in distribution holds for all $t \ge 0$ (jointly):

$$
Q_v(-t) \stackrel{d}{=} \sup_{s \ge t} X(s) - X(t).
$$

The challenge in simulating $Q_v(-t)$ involves sampling $M(t) = \max_{s \ge t}\{X(s)\}$ jointly with $X(t)$ during any time interval of the form $[0, T]$ for $T > 0$. The rest of the section is devoted to solve this challenge.

The idea is to identify a sequence of random times Δ_k such that

$$
\max_{T_k^0 \le t \le T_k^0 + \Delta_k} \{X(t)\} \ge \max_{t \ge T_k^0 + \Delta_k} \{X(t)\}.
$$

Then, $M(T_k^0) = \max_{t \ge T_k^0}\{X(t)\} = \max_{T_k^0 \le t \le T_k^0 + \Delta_k}\{X(t)\}$. In particular, to calculate $M(T_k^0)$, we only need to look at the maximum of $X(t)$ over a *finite* time

interval, $[T_k^0, T_k^0 + \Delta_k]$. To find such Δ_k, we apply two tricks here. The first trick is to decompose $X(t)$ into $(c+1)$ random walks with negative drift associated with N^i for $i = 0, 1, \ldots, c$. This is based on the fact that for $\lambda < c\mu$, we can pick $a \in (\lambda, c\mu)$ such that $N^0(t) - at$ and $\left((a/c)t - N^i(t)\right)$ are "drifting downward" to negative infinity. We can then bound $M(t)$ by the "corresponding" running time maximum of the random walks with negative drift. The second trick is a "milestone event" construction, which allows us to identify random times beyond which a random walk with negative drift will never go above a previously achieved level.

The "milestone events" are similar to the ladder height decomposition of a random walk, but we cannot directly use ladder height theory because the corresponding expressions for the probabilities of interest (for example the probability of an infinite strictly increasing ladder epoch) are rarely computable in closed form. The "milestone construction" introduces a parameter m which, together with change of measure ideas, allows us to simulate without bias the occurrence of object such as the time the random walk reaches a certain barrier, for example.

Putting these "milestone events" of the random walks together and using the fact that $M(t)$ can be bounded by the appropriate running time maximums of the random walks, we can find the desired Δ_k. We next provide the details of the construction.

Decomposition Choose $a \in (\lambda, c\mu)$. Then, for $t > 0$,

$$X(t) = N^0(t) - \sum_{i=1}^{c} N^i(t) = (N^0(t) - at) + \sum_{i=1}^{c} \left(\frac{a}{c}t - N^i(t)\right).$$

We define $(c+1)$ random walks with negative drift associated with $N^i(t)$'s as follows:

$$S_0^{(0)} = 0, \quad S_1^{(0)} = -aT_1^0 + 1, \quad S_n^{(0)} = S_{n-1}^{(0)} + (-aA_{n-1} + 1) \text{ for } n \geq 2. \tag{16}$$

In particular, $S_n^{(0)} = N^0(T_n^0) - aT_n^0$. For $i = 1, \ldots, c$,

$$S_0^{(i)} = \frac{a}{c}T_1^i, \quad S_n^{(i)} = S_{n-1}^{(i)} + \left(-1 + \frac{a}{c}V_n^i\right) \text{ for } n \geq 1. \tag{17}$$

Here, $S_n^{(i)} = N^i(T_n^i-) - aT_n^i$. Figure 2 plots the relationship between $\{N^0(t) - at : t \geq 0\}$ and $\{S_n^{(0)} : n \geq 0\}$, and the relationship between $\{\frac{a}{c}t - N^i(t) : t \geq 0\}$ and $\{S_n^{(i)} : n \geq 0\}$ for $i = 1, \ldots, c$. In particular, we notice from Fig. 2 that

$$\max_{s \geq t}\{N^0(s) - as\} = \max\left\{N^0(t) - at, \max_{n \geq N^0(t)+1}\left\{S_n^{(0)}\right\}\right\} \leq \max_{n \geq N^0(t)}\left\{S_n^{(0)}\right\},$$

and for $i = 1, \ldots, c$,

$$\max_{s \geq t}\left\{\frac{a}{c}s - N^i(s)\right\} = \max_{n \geq N^i(t-)}\left\{S_n^{(i)}\right\}. \leq \max_{n \geq (N^i(t)-1)^+}\left\{S_n^{(i)}\right\}.$$

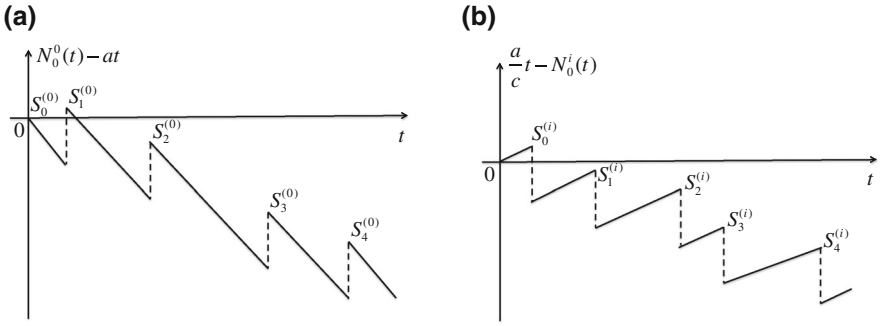


Fig. 2 The relationship between the renewal processes and the random walks **a** $N^0(t) - at$ and $S_n^{(0)}$ **b** $(a/c)t - N^i(t)$ and $S_n^{(i)}$

We then notice that, for any given T,

$$\begin{aligned} M(T) = \max_{t\geq T}\{X(t)\} &= \max_{t\geq T}\left\{(N^0(t)-at)+\sum_{i=1}^{c}\left(\frac{a}{c}t-N^i(t)\right)\right\} \\ &\leq \max_{t\geq T}\left\{N^0(t)-at\right\}+\sum_{i=1}^{c}\max_{t\geq T}\left\{\frac{a}{c}t-N^i(t)\right\} \\ &\leq \max_{n\geq N^0(T)}\left\{S_n^{(0)}\right\}+\sum_{i=1}^{c}\max_{n\geq N^i(T)-1}\left\{S_n^{(i)}\right\}. \end{aligned} \tag{18}$$

Milestone construction We use the "milestone events" construction to generate the $(c+1)$ random walks with negative drift, $S^{(i)}$, together with their running time maxima, $M_k^{(i)} := \max_{n\geq k}\{S_n^{(i)}\}$, $k \geq 0$, $i = 0, 1, \ldots c$. This construction is introduced in [5,6], and we shall provide a brief overview here.

Fix $m > 0$ and $L \geq 1$ such that $P(m < M_0^{(i)} \leq (L+1)m) > 0$ for $i = 0, \ldots, c$. The values of m and L do not seem to have significant impact on algorithm performance, as long as they are chosen to be small. In our numerical implementation, we choose $m = 1$ and $L = 3$.

For each random walk $\{S_n^{(i)} : n \geq 0\}$, $i = 0, 1, \ldots, c$, we shall define a sequence of downward and upward "milestone events," which we denoted as Φ_j^i and Υ_j^i, respectively, for $j \geq 0$ as follows:

$$\Phi_0^i := 0, \quad \Upsilon_0^i := 0,$$

and for $j \geq 1$,

$$\begin{aligned} \Phi_j^i &:= \inf\left\{n \geq \Upsilon_{j-1}^i I(\Upsilon_{j-1}^i < \infty) \vee \Phi_{j-1}^i : S_n^{(i)} < S_{\Phi_{j-1}^i}^{(i)} - Lm\right\}, \\ \Upsilon_j^i &:= \inf\left\{n \geq \Phi_j^i : S_n^{(i)} > S_{\Phi_j^i}^{(i)} + m\right\}. \end{aligned}$$

Notice that $P(\Phi_j^i < \infty) = 1$ while $P(\Upsilon_j^i < \infty) < 1$, as the random walks have negative drift. In fact, under (A1), Proposition 2.1 in [6] shows $P(\Upsilon_j^i = \infty,\ i.o.) = 1$. We observe that when the event $\{\Upsilon_j^i = \infty\}$ happens, we know that the random walk will never go above $S^{(i)}_{\Phi_j^i} + m$ beyond Φ_j^i. This important observation allows us to find the running time maximum $M_k^{(i)}$. In particular, let Φ_{k*}^i denote the first downward milestone at or after step k, and let Φ_{k**}^i be the first downward milestone after Φ_{k*}^0 with $\Upsilon_{k**}^i = \infty$. Then, after step Φ_{k**}^i, the random walk $S^{(i)}$ will never go above the level $S^{(i)}_{\Phi^0_{k**}} + m$, and $S^{(i)}_{\Phi^0_{k**}} + m < S^{(i)}_{\Phi^0_{k*}} - Lm + m \le S^{(i)}_{\Phi^i_{k*}}$. Therefore, $M_k^{(i)} = \max_{n\ge k}\{S_n^{(i)}\} = \max_{k\le n\le \Phi^i_{k**}}\{S_n^{(i)}\}$, i.e., we just need to find the maximum value of the random walk between step k and step Φ^i_{k**}. Figure 3 provides a pictorial explanation of the construction.

We are now ready to use the milestone events across the $(c+1)$ random walks to identify Δ_k associated with each T_k^0 $(k \ge 1)$, such that $N^i(T_k^0) \ge 1$ for $i = 1, \ldots, c$. Define

$$\Lambda_k^0 := \min_{j\ge 1}\left\{\Phi_j^0 > N^0\left(T_k^0\right) : S^{(0)}_{\Phi_j^0} \le S^{(0)}_{N^0(T_k^0)} - m,\ \Upsilon_j^0 = \infty\right\}, \tag{19}$$

and, for $i = 1, \ldots, c$,

$$\Lambda_k^i := \min_{j\ge 1}\left\{\Phi_j^i > N^i\left(T_k^0\right) - 1 : S^{(i)}_{\Phi_j^i} \le S^{(i)}_{N^i(T_k^0)-1} - m,\ \Upsilon_j^i = \infty\right\}. \tag{20}$$

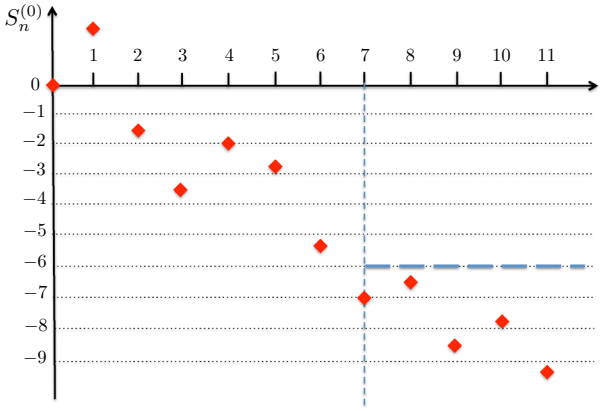


Fig. 3 This figure plots a realization of the sample path $\{S_n^{(0)} : 0 \le n \le 11\}$. Here, we set $m = 1$ and $L = 3$. Then, $\Phi_1^0 = 3$, $\Upsilon_1^0 = 4$, $\Phi_2^0 = 7$. If $\Upsilon_2^0 = \infty$, then for $n \ge 7$, $S_n^{(0)}$ will stay below the level $S_7^{(0)} + m$, which is demonstrated by the bold dashed line. Thus, $M_2^{(0)} = \max_{n\ge 2}\{S_n^{(0)}\} = S_2^{(0)}$ by only comparing the random walk values between step 2 and step 7

In particular, the random walk $\{S_n^{(i)} : n \geq 0\}$ will never go above the level $S^{(i)}_{\Lambda_k^i} + m$ for $n \geq \Lambda_k^i$, $i = 0, \ldots, c$. Let

$$\Delta_k := \max\left\{T^0_{\Lambda_k^0}, \max_{1\leq i\leq c}\left\{T^i_{\Lambda_k^i+1}\right\}\right\} - T_k^0. \tag{21}$$

Since $N^0\left(T_k^0 + \Delta_k\right) \geq \Lambda_k^0$ and $N^i(T_k^0 + \Delta_k) - 1 \geq \Lambda_k^i$ for $i = 1, \ldots, c$,

$$\max_{n\geq N^0\left(T_k^0+\Delta_k\right)}\left\{S_n^{(0)}\right\} \leq S^{(0)}_{\Lambda_k^0} + m \quad \text{and} \quad \max_{n\geq N^i\left(T_k^0+\Delta_k\right)-1}\left\{S_n^{(i)}\right\} \leq S^{(i)}_{\Lambda_k^i} + m.$$

Therefore,

$$\begin{aligned}
\max_{t\geq T_k^0+\Delta_k}\{X(t)\} &\leq \max_{n\geq N^0\left(T_k^0+\Delta_k\right)}\left\{S_n^{(0)}\right\} + \sum_{i=1}^{c}\max_{n\geq N^i\left(T_k^0+\Delta_k\right)-1}\left\{S_n^{(i)}\right\} \\
&\leq S^{(0)}_{\Lambda_k^0} + m + \sum_{i=1}^{c}\left(S^{(i)}_{\Lambda_k^i} + m\right) \\
&\leq S^{(0)}_{N^0\left(T_k^0\right)} + \sum_{i=1}^{c} S^{(i)}_{N^i\left(T_k^0\right)-1} \\
&\leq N^0\left(T_k^0\right) - aT_k^0 + \sum_{i=1}^{c}\left(\frac{a}{c}T_k^0 - N^i\left(T_k^0\right)\right) \\
&= X\left(T_k^0\right) \leq \max_{T_k^0\leq t\leq T_k^0+\Delta_k}\{X(t)\}.
\end{aligned}$$

Under (A1), the time it takes to find Δ_k using the "milestone" construction has finite expectation (Theorem 2.2 in [6]). We shall provide the algorithmic details to generate the random walk with negative drift together with the "milestone" events for the light-tailed case in Sect. 4.1 to demonstrate the basic idea. The general case can be found in [6]. We also provide the algorithm to match the service time requirements to the customers in the vacation system between two consecutive inspection times in Sect. 4.2. Lastly, the exact simulation algorithm of the *GI/GI/c* queue is summarized in Sect. 4.3.

4.1 Simulate a random walk with negative drift jointly with "milestone" events

To demonstrate the basic idea, we work with a generic random walk with negative drift $S_n := S_{n-1} + X_n$, for $n \geq 0$, with S_0 given. We also impose the light-tail assumption on X_n, i.e., there exist $\theta > 0$ such that $E[\exp(\theta X_n)] < \infty$. Let

$$\Phi_0 := 0, \quad \Upsilon_0 := 0,$$

and, for $j \geq 1$,

$$\Phi_j := \inf\left\{n \geq \Upsilon_{j-1} I(\Upsilon_{j-1} < \infty) \vee \Phi_{j-1} : S_n < S_{\Phi_{j-1}} - Lm\right\},$$
$$\Upsilon_j := \inf\left\{n \geq \Phi_j : S_n > S_{\Phi_j} + m\right\}.$$

We also denote $\tau_m = \inf\{n \geq 0 : S_n > m, S_0 = 0\}$. Notice that $P(\Upsilon_j = \infty) = P(\tau_m = \infty) > 0$.

Sampling Φ_j is straightforward. We just sample the random walk, S_n, until $S_n < S_{\Phi_{j-1}} - Lm$. Sampling Υ_j and the path conditional on $\Upsilon_j < \infty$ require more advanced simulation techniques, as $P(\Upsilon_j = \infty) > 0$. In particular, we use the exponential tilting idea discussed in [1]. Let $\psi_X(\theta) = \log E\left[\exp(\theta X_n)\right]$ be the log moment generating function of X_n, then we have $EX_n = \psi'_X(0) < 0$ and $Var(X_n) = \psi''_X(0) > 0$. By the convexity of $\psi_X(\cdot)$, we can always find $\eta > 0$ with $\psi_X(\eta) = 0$ and $\psi'_X(\eta) \in (0, \infty)$. Hence, we can define a new measure P_η based on exponential tilting so that

$$\frac{\mathrm{d}P_\eta}{\mathrm{d}P}(X_n) = \exp(\eta X_n).$$

Under P_η, S_n is a random walk with positive drift $\psi'_X(\eta)$. Thus, $P_\eta(\tau_m < \infty) = 1$. By our choice of η, we also have $P(\tau_m < \infty) = E_\eta \exp(-\eta S_{\tau_m})$. In implementation, we shall generate the path S_n under P_η until τ_m and check whether $U \leq \exp(-\eta S_{\tau_m})$, where U is a uniform random variable independent of everything. If $U \leq \exp(-\eta S_{\tau_m})$, we claim that $\tau_m < \infty$ and accept the path $(S_n : n \leq \tau_m)$ as the path of the random walk conditional on $\tau_m < \infty$.

The algorithm to sample the random walk together with the milestone events goes as follows. *Throughout this paper, "sample" in the pseudocode means sampling independently from everything that has already been sampled.*

Algorithm RWS: Sample a random walk with negative drift until stopping criteria are met
Input: $L, m, \{S_0, \ldots, S_n\}, \{\Phi_0, \ldots, \Phi_j\}, \{\Upsilon_0, \ldots, \Upsilon_j\}$ and stopping criteria $\mathcal{H}$.
(Note that $n = \Phi_j$ if $\Upsilon_j = \infty$, and $n = \Upsilon_j$ otherwise. If there is no previous simulated partial random walk, then we initialize $n = 0$, $j = 0$, $\Phi_0 = 0$, $\Upsilon_0 = 0$, and S_0 as needed.)

1. While the stopping criteria $\mathcal{H}$ are not satisfied, set $j \leftarrow j + 1$.
   (a) (Downward milestone simulation)
   Sample $\{S_k : n + 1 \leq k \leq \Phi_j\}$ under the nominal measure, i.e., generate the random walk until $S_n < S_{\Phi_{j-1}} - Lm$. Update $n = \Phi_j$.
   (b) (Upward milestone simulation)
   Sample $\tilde{S}_1, \ldots, \tilde{S}_{\tau_m}$ from the tilted measure $P_\eta(\cdot)$. Sample $U \sim$ Uniform[0,1]. If $U \leq \exp\left(-\eta \tilde{S}_{\tau_m}\right)$, set $\Upsilon_j = n + \tau_m$, $S_{n+k} = S_n + \tilde{S}_k$ for $k = 1, \ldots, \tau_m$ and update $n \leftarrow n + \tau_m$; otherwise set $\Upsilon_j = \infty$.
2. Output updated $\{S_0, \ldots, S_n\}, \{\Phi_0, \ldots, \Phi_j\}$ and $\{\Upsilon_0, \ldots, \Upsilon_j\}$.

4.2 Simulate the vacation system between inspection times

To summarize our discussion above, in this section, we provide the pseudocode for generating the vacation system between the inspection time $T^0_{\kappa_l}$ and $T^0_{\kappa_{l+1}}$, for $l \geq 0$, $\kappa_0 = 0$, and $\kappa_{l+1} < \kappa_l$.

Algorithm VSS: sample vacation system between $T^0_{\kappa_l}$ and $T^0_{\kappa_{l-1}}$, and extract corresponding service times

Input: $m, L, \kappa_l, \kappa_{l-1}, \{S^{(i)}_0, \ldots, S^{(i)}_{n_i}\}, \{\Phi^i_0, \ldots, \Phi^i_{j_i}\}, \{\Upsilon^i_0, \ldots, \Upsilon^i_{j_i}\}$ for $i = 0, 1, \ldots, c$.

1. Apply Algorithm RWS to further sample $S^{(0)}$ with the stopping criterion $\mathcal{H}$ being $n_0 \geq |\kappa_l|$. Then, find $T^0_{|\kappa_l|}$.
2. Apply Algorithm RWS to further sample $S^{(0)}$ with the stopping criterion $\mathcal{H}$ being $n_0 = \Lambda^0_{|\kappa_l|}$, with $\Lambda^0_{|\kappa_l|}$ defined in Eq. (19).
3. For $i = 1, \ldots, c$, apply Algorithm RWS to further sample $S^{(i)}$ until the stopping criterion $\mathcal{H}$ being $n_i = \Lambda^i_{|\kappa_l|}$, with $\Lambda^i_{|\kappa_l|}$ defined in Eq. (20).
4. Compute $\Delta_{|\kappa_l|}$ as defined in Eq. (21). For $i = 0, 1, \ldots, c$, apply Algorithm RWS to further sample $S^{(i)}$ with the stopping criterion $\mathcal{H}$ being $T^i_{n_i} \geq T^0_{|\kappa_l|} + \Delta_{|\kappa_l|}$.
5. Construct the backward renewal processes $\{N^i(t) : T^0_{\kappa_l} - \Delta_{|\kappa_l|} \leq t \leq 0\}$ using $\{S^{(i)}_n : 0 \leq n \leq n_i\}$ for $i = 0, 1, \ldots, c$. In particular, we shall set $T^i_{-n} = -T^i_n$. Then, construct $X(t) = N^0(t) - \sum_{i=1}^c N^i(t)$ for $t \in [T^0_{\kappa_l} - \Delta_{|\kappa_l|}, 0]$.
6. Set $M(T^0_{\kappa_l}) = \max_{T^0_{\kappa_l} - \Delta_{|\kappa_l|} \leq t \leq T^0_{\kappa_l}} \{X(t)\}$ and then compute $Q_v(T^0_{\kappa_l}) = M(T^0_{\kappa_l}) - X(T^0_{\kappa_l})$ to be the number of people waiting in the queue at time $T^0_{\kappa_l}$. The remaining activity times are $U^i(T^0_{\kappa_l})$, for $i = 1, \ldots, c$.
7. If $Q_v(T^0_{\kappa_l}) > 1$, then for $1 \leq j \leq Q_v(T^0_{\kappa_l}) - 1$, the j-th people waiting in queue arrive at time $T^0_{\kappa_l - Q_v(T^0_{\kappa_l}) + j}$. Let $\tilde{D}_j = \inf\{t \geq 0 : j = \sum_{i=1}^c \sigma^i_{T^0_{\kappa_l}}(t)\}$, then extract $V_{\kappa_l - Q_v(T^0_{\kappa_l}) + j} = \sum_{i=1}^c V^i_{N^i(T^0_{\kappa_l} + \tilde{D}_j)} \mathrm{d}N^i(T^0_{\kappa_l} + \tilde{D}_j)$ as his service time.
8. For $\kappa_l \leq n \leq -1$, use Eq. (3) to extract the service times $V_{\kappa_l}, \ldots, V_{-1}$.
9. Output
   (a) service times of the people waiting in queue at time $T^0_{\kappa_l}$ (excluding the arrival at $T^0_{\kappa_l}$), i.e., null if $Q_v(T^0_{\kappa_l}) = 1$ and $(V_{\kappa_l - Q_v(T^0_{\kappa_l}) + 1}, \ldots, V_{\kappa_l - 1})$ in the order of arrival if $Q_v(T^0_{\kappa_l}) > 1$.
   (b) matched arrival times and service times $\left\{\left(T^0_j, V_j\right) : \kappa_l \leq j \leq -1\right\}$ in the order of arrival.
   (c) updated random walks $\{S^{(i)}_0, \ldots, S^{(i)}_{n_i}\}$ with updated milestone events $\{\Phi^i_0, \ldots, \Phi^i_{j_i}\}, \{\Upsilon^i_0, \ldots, \Upsilon^i_{j_i}\}$ for $i = 0, 1, \ldots, c$.

4.3 Overall exact simulation procedure

In this section, we provide the overall pseudocode for our exact simulation algorithm.

Algorithm PS: sample stationary *GI/GI/c* queue at time 0

Input: m, L, F, G, c

1. For $i = 0, 1, \ldots, c$, initiate $\Phi_0^i = \Upsilon_0^i = 0$, and $S_0^{(i)}$ as defined in Eqs. (16, 17).
2. Set $\kappa_0 = 0, \kappa_1 = -10, l = 1$.
3. (a) Apply Algorithm VSS to sample vacation system between $T_{\kappa_l}^0$ and $T_{\kappa_{l-1}}^0$, and extract corresponding service times.
   (b) Start two *GI/GI/c* queues, both from $T_{\kappa_l}^0$, one initialized with

$$\left(Q_v\left(T_{\kappa_l}^0\right), \mathcal{S}\left(U\left(T_{\kappa_l}^0\right)\right), 0\right)$$

   and the other initialized with $\mathbf{0}$. Evolve the two queues forward in time until time 0 and calculate

$$C = \min_{\kappa_l \le j \le -1} \left\| Z_{T_{\kappa_l}^0}\left(T_j^0 - T_{\kappa_l}^0; \left(Q_v\left(T_{\kappa_l}^0\right), \mathcal{S}\left(U\left(T_{\kappa_l}^0\right)\right), 0\right)\right) - Z_{T_{\kappa_l}^0}\left(T_j^0 - T_{\kappa_l}^0; \mathbf{0}\right)\right\|_\infty .$$

4. If $C = 0$, output $Z(0) = Z_{T_{\kappa_l}^0}\left(|T_{\kappa_l}^0|; \mathbf{0}\right)$. Otherwise ($C > 0$), set $l \leftarrow l + 1$, $\kappa_l = 2\kappa_{l-1}$, then go back to Step 3.

5 Numerical experiments

As a sanity check, we have implemented our MATLAB code in the case of an Erlang (k_1, λ)/Erlang$(k_2, \mu)/c$ queue. We provide a detailed MATLAB implementation of each of the algorithms required to execute Facts I–IV in the online Appendix to this paper.

Firstly, in the context of the *M/M/c* queue, which is a special case of Erlang(k_1, λ)/Erlang$(k_2, \mu)/c$ when $k_1 = k_2 = 1$ and whose stationary distribution can be computed in closed form, we have compared the theoretical distribution to the empirical distribution of the number of customers in the system at stationarity. The empirical distribution is produced from a large number of runs using our perfect simulation algorithm. Figure 4 shows a comparison of these distributions when $\lambda = 3$, $\mu = 2$, and $c = 2$. Gray bars show the empirical result of 5000 draws using our perfect simulation algorithm, and black bars show the theoretical distribution of the number of customers in the system. The two are very close to each other. Following [9], we test the goodness of fit using a Pearson's chi-squared test; under the null hypothesis, the empirical histogram converges to theoretical distribution as the sample size increases. The test yields a p value equal to 0.6806, indicating close agreement (i.e., we can not reject the null hypothesis). Similarly, Fig. 5 provides another comparison with a different set of parameters, $\lambda = 10$, $\mu = 2$, $c = 10$, with a p value being 0.6454 from the chi-squared test.

Also, for a general Erlang(k_1, λ)/Erlang$(k_2, \mu)/c$ queue ($k_1 > 1, k_2 > 1$) when $\rho = (\lambda_1 k_2)/(c\lambda_2 k_1) = 0.9$, we have compared the empirical distribution obtained from simulation with the numerical results (with precision at least 10^{-4}) provided in Table III of [17]. Figure 6 shows the comparison for an $E_3/E_2/5$ queue with $\rho = 0.9$. We observe that the two histograms are very close to each other. A Pearson's chi-

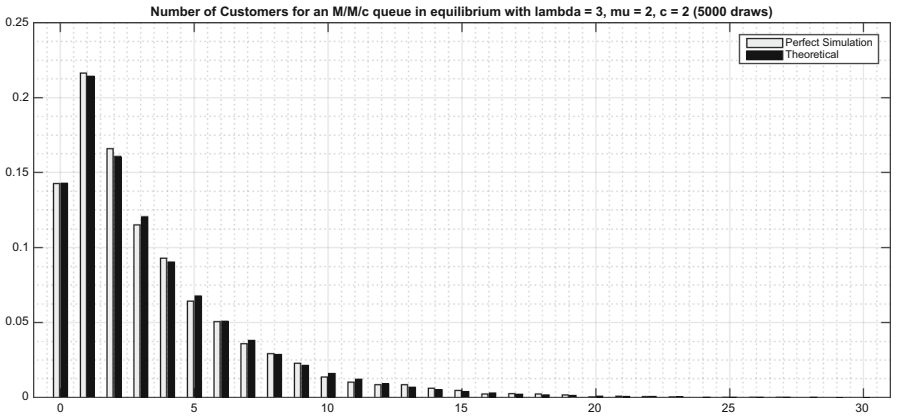


Fig. 4 Number of customers for an *M/M/c* queue in stationarity when $\lambda = 3$, $\mu = 2$, and $c = 2$

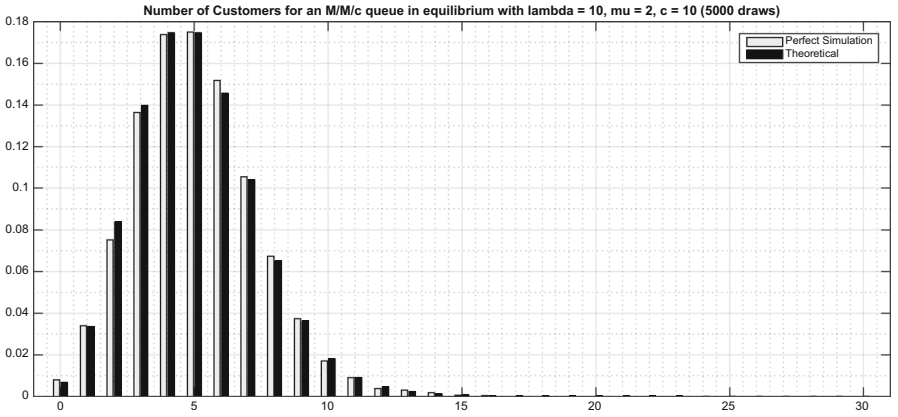


Fig. 5 Number of customers for an *M/M/c* queue in stationarity when $\lambda = 10$, $\mu = 2$, and $c = 10$

squared test between the simulated distribution and the numerical one gives a p value of 0.6815.

Next, we run numerical experiments in the *M/M/c* case to see how the running time of our algorithm, measured by mean coalescence time of two bounding systems, scales as the number of servers grows and the traffic intensity ρ changes. Starting from time 0, the upper bound queue has its queue length sampled from the theoretical distribution of an *M/M/c* vacation system and all servers busy with remaining service times drawn from the equilibrium distribution of the service/vacation time; and the lower bound queue is empty. Then, we run both the upper bound and lower bound queues forward in time with the same stream of arrival times and service requirements until they coalesce. Table 1 shows the estimated average coalescence time, $E[T]$, based on 5000 iid samples, for different system scales in the quality-driven regime (QD). We observe that $E[T]$ does not increase much as the system scale parameter, s, grows. Table 2 shows similar results for the quality-and-efficiency-driven operating

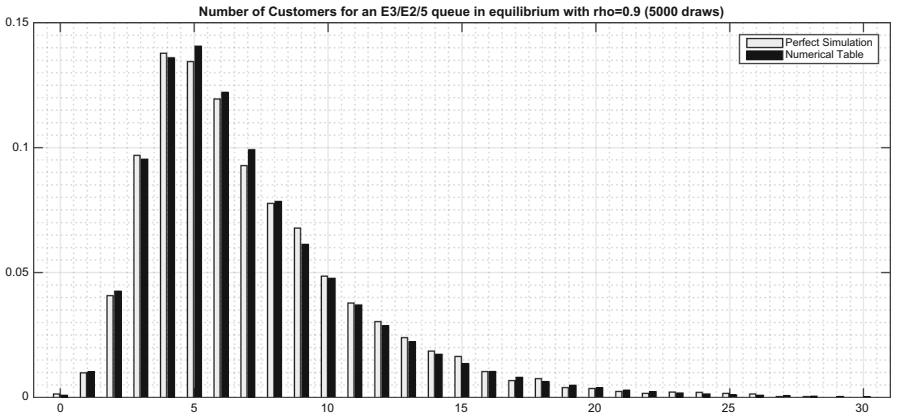


Fig. 6 Number of customers for an Erlang(k_1, λ)/Erlang$(k_2, \mu)/c$ queue in stationarity when $k_1 = 3$, $\lambda = 4.5$, $k_2 = 2$, $\mu = 2/3$, $c = 5$, and $\rho = 0.9$

Table 1 Simulation result for coalescence time of *M/M/c* queue

s	Mean	95% confidence interval
(QD: $\lambda_s = s, c_s = 1.2s, \mu = 1$)		
100	6.4212	[6.2902, 6.5522]
500	7.0641	[6.9848, 7.1434]
1000	7.7465	[7.6667, 7.8263]

Table 2 Simulation result for coalescence time of *M/M/c* queue

s	Mean	95% confidence interval
(QED: $\lambda_s = s, c_s = s + 2\sqrt{s}, \mu = 1$)		
100	6.5074	[6.3771, 6.6377]
500	8.5896	[8.4361, 8.7431]
1000	9.4723	[9.3041, 9.6405]

regime (QED). In this case, $E[T]$ increases at a faster rate with s than the QD case, but the magnitude of increment is still not significant.

Finally, we run a numerical experiment in the *M/M/c* case aiming to test how computational complexity of our algorithm changes with traffic intensity, $\rho = \lambda/c\mu$. Here, we define the computational complexity as the total number of renewals (including arrivals and services/vacations) the algorithm samples in total to find the coalescence time. We expect the complexity to scale like $(c+1)(1-\rho)^{-2}E[T(\rho)]$, where $(c+1)$ is the number of renewal processes we need to simulate, $(1-\rho)^{-2}$ is on average the amount of renewals we need to sample to find its running time maximum for each renewal process, and $E[T(\rho)]$ is the mean coalescence time when the traffic intensity is ρ. Table 3 summarizes our numeral results, based on 5000 independent runs of the algorithm for each ρ. We run the coalescence check at 10×2^k for $k = 1, 2, \ldots$, until we find the coalescence. We observe that as ρ increases, the computational complexity increases significantly, but when multiplied by $(1-\rho)^2$, the resulting products are of

Table 3 Simulation result for computational complexities with varying traffic intensities

λ	Traffic intensity (ρ)	Mean number of renewals sampled	Mean index of successful inspection time	Mean number of renewals sampled $\times (1-\rho)^2$
M/M/c queue with fixed $\mu = 5$ and $c = 2$				
5	0.5	225.6670	11.7780	56.4168
6	0.6	377.0050	14.7780	60.3208
7	0.7	764.3714	21.9800	68.7934
8	0.8	2181.3452	44.2320	87.2538
9	0.9	12,162.6158	161.0840	121.6262

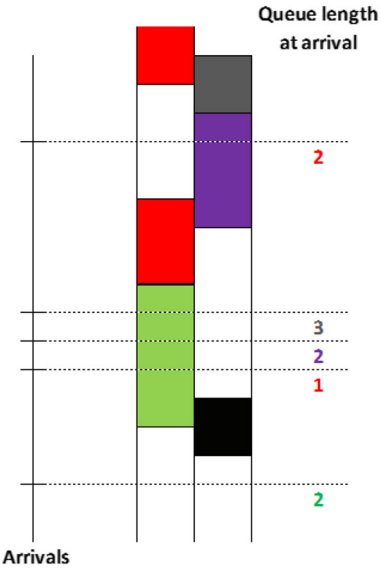


Fig. 7 Matching procedure of service times to arrival process

about the same magnitude—up to a factor proportional to λ, given that the number of arrivals scales as λ per unit time. Therefore, the main scaling parameter for the complexity here is $(1-\rho)^{-2}$. Notice that if we simulate the system forward in time from empty, it also took around $O\left((1-\rho)^{-2}\right)$ arrivals to get close to stationary.

6 Proof of technical results

6.1 The iid property of the coupled service times and independence of the arrival process

In order to explain why the V_n form an iid sequence, independent of the sequence $\mathcal{T}^0 = \{T_n^0 : n \in \mathbb{Z}\backslash\{0\}\}$, it is useful to keep in mind the diagram depicted in Fig. 7, which illustrates a case involving two servers, $c = 2$.

The assignment of the service times, as we shall explain, can be thought of as a procedure similar to a Tetris game. Arrival times are depicted by dotted horizontal lines which go from left to right, starting at the left most vertical line, which is labeled "Arrivals." Think of the time line going, vertically, from the bottom of the graph (past) to the top of the graph (future).

In the right-most column in Fig. 7, we indicate the queue length, right at the time of a depicted arrival (and thus, including the arrival itself). So, for example, the first arrival depicted in Fig. 7 observes one customer waiting and thus, including the arrival himself, there are two customers waiting in queue.

The Tetris configuration observed by an arrival at time T is comprised of two parts: (i) the receding horizon, which corresponds to the remaining incomplete blocks, and (ii) the landscape, comprised of the configuration of complete blocks. So, for example, the first arrival in Fig. 7 observes a receding horizon corresponding to the two white remaining blocks, which start from the dotted line at the bottom. The landscape can be parameterized by a sequence of block sizes, and the order of the sequence is given by the way in which the complete blocks appear from bottom to top—this is precisely the Tetris-game assignment. There are no ties because of the continuous time stationarity and independence of the underlying renewal processes. The colors are, for the moment, not part of the landscape. We will explain the meaning of the colors momentarily.

The assignment of the service times is done as follows: The arriving customer reads off the right-most column (with heading "Queue length at arrival") and selects the block size labeled precisely with the number indicated by the "Queue length at arrival." So, there are two distinctive quantities to keep in mind assigned to each player (i.e., arriving customer): (a) the landscape (or landscape sequence, which, as indicated, can be used to reconstruct the landscape), and (b) the *service time*, which is the complete block size occupying the "Queue length at arrival"-th position in the landscape sequence.

The color code in Fig. 7 simply illustrates quantity b) for each of the arrivals. So, for example, the first arrival, who reads "Queue length at arrival = 2" (which we have written in green color), gets assigned the second complete block, which we have depicted in green. Similarly, the second arrival depicted reads off the number "1" (written in red) and gets assigned the first red block depicted (from bottom to top). The very first complete block (from bottom to top), which is depicted in black, corresponds to the service time assigned to the customer ahead of the customer who collected the green block. The number "1" (in red) is obtained by observing that the customer with the initial black block has departed.

Now we argue the following properties:

(1) The service times are iid copies of V.
(2) The service times are independent of $\mathcal{T}^0$.

About property (1): The player arriving at time T reads a number, corresponding to the queue length, which is obtained by the *past filtration* $\mathcal{F}_T$ generated by $\cup_{k\in\mathbb{Z}\setminus\{0\},0\le i\le c}\{T_k^i : T_k^i \le T\}$. Conditional on the receding horizon (i.e., remaining incomplete block sizes), $\mathcal{R}_T$, the past filtration is independent of the landscape. This is simply the Markov property applied to the forward residual lifetime process of each of the c renewal processes represented by the c middle columns. Moreover, conditional

on $\mathcal{R}_T$, each landscape forms a sequence of iid copies of V because of the structure of the underlying c renewal processes corresponding to the middle columns. So, let $Q(T)$ denote the queue length at time T (including the arrival at time T), which is a function of the past filtration, and let $\{L_T(k) : k \geq 1\}$ be the landscape sequence observed at time T, so that $L_T(Q(T))$ is the service time of the customer who arrives at time T. We then have that, for any positive and bounded continuous function $f(\cdot)$,

$$E[f(L_T(Q(T)))|\mathcal{R}_T] = E[f(L_T(1))|\mathcal{R}_T] = E[f(V)],$$

precisely because, conditional on $\mathcal{R}_T$, $Q(T)$ (being $\mathcal{F}_T$ measurable) is independent of $\{L_T(k) : k \geq 1\}$.

To verify the iid property, let f_1, f_2 be non-negative and bounded continuous functions. Assume that $T_1 < T_2$ are arrival times in $\mathcal{T}^0$ (not necessarily consecutive). Then,

$$\begin{aligned}
&E\left[f_1\left(L_{T_1}(Q(T_1))\right) f_2\left(L_{T_2}(Q(T_2))\right)\right] \\
&\quad = E\left[E\left[f_1\left(L_{T_1}(Q(T_1))\right) f_2\left(L_{T_2}(Q(T_2))\right)|\mathcal{F}_{T_2}, \mathcal{R}_{T_2}\right]\right] \\
&\quad = E\left[f_1\left(L_{T_1}(Q(T_1))\right) E\left[f_2\left(L_{T_2}(Q(T_2))\right)|\mathcal{F}_{T_2}, \mathcal{R}_{T_2}\right]\right] \\
&\quad = E\left[f_1(L_{T_1}(Q(T_1)))\right] E[f_2(V)] = E[f_1(V)]E[f_2(V)].
\end{aligned}$$

The same argument extends to any subset of arrival times, and thus the iid property follows.

About property (2): Note that in the calculations involving property (1), the actual values of the arrival times T, T_1, and T_2 are irrelevant. The iid property of the service times is established path-by-path conditional on the observed realization $\mathcal{T}^0$. Thus, the independence of the arrival process and service times follows immediately.

6.2 Proof of technical lemmas of monotonicity

Proof of Lemma 1 Both facts are standard; the first one can be easily shown using induction. Specifically, we first notice that $W_k(T_k^0; w^+) = w^+ > w^- = W_k(T_n^0; w^-)$. Suppose that $W_k(T_n^0; w^+) \geq W_k(T_n^0; w^-)$ for some $n \geq k$, then

$$\begin{aligned}
W_k\left(T_n^{0,+}; w^+\right) &= \mathcal{S}\left(\left(W_k\left(T_n^0; w^+\right) + V_n\mathbf{e_1} - A_n\mathbf{1}\right)^+\right) \\
&\geq \mathcal{S}\left(\left(W_k\left(T_n^0; w^-\right) + V_n\mathbf{e_1} - A_n\mathbf{1}\right)^+\right) = W_k\left(T_n^{0,+}; w^-\right).
\end{aligned}$$

For inequality (9), we note that $W_k\left(T_{k'}^0; 0\right) \geq W_{k'}\left(T_{k'}^0; 0\right) = 0$, and therefore, due to (8), we have that

$$W_k\left(T_n^0; 0\right) = W_{k'}\left(T_n^0; W_k\left(T_{k'}; 0\right)\right) \geq W_{k'}\left(T_n^0; 0\right).$$

□

Proof of Lemma 2 This fact follows immediately by induction from Eqs. (4) and (7) using the fact that $\Xi_n \geq 0$. □

Proof of Lemma 3 We first prove the inequality $Q_u(t-u; z^+) \leq Q_v(t)$. Note that $U^i(u) > 0$ for all u (the forward residual lifetime process is right continuous), so the initial condition r indicates that all the servers are busy (operating) and the initial $q \geq 0$ customers will leave the queue (i.e., enter service) at the same time as those in the vacation system under the evolution of $Z_u\left(\cdot; z^+\right)$. Now, let us write $N = \inf\{n : T_n^0 \geq u\}$ (in words, the next arriving customer at or after u arrives at time T_N^0). It is easy to see that $\mathcal{S}\left(U\left(T_N^0\right)\right) \geq R_u(T_N^0 - u; z^+)$; to wit, if T_N^0 occurs before any of the servers becomes idle, then we have equality, and if T_N^0 occurs after, say, $l \geq 1$ servers become idle, then $R_u(T_N^0 - u; z^+)$ will have l zeroes and the bottom $c-l$ entries will coincide with those of $\mathcal{S}\left(U\left(T_N^0\right)\right)$, which has strictly positive entries. So, if w_N is the Kiefer–Wolfowitz vector observed by the customer arriving at T_N^0 (induced by $Q_u(\cdot - u; z^+)$), then we have $W_v(N) \geq w_N$. By monotonicity of the Kiefer–Wolfowitz vector in the initial condition and because of Lemma 2, we have

$$W_v(k) \geq W_N\left(T_k^0; W_v(N)\right) \geq W_N\left(T_k^0; w_N\right),$$

for all $k \geq N$, and hence, $T_k^0 + D_k^0 \geq T_k^0 + W_N^{(1)}\left(T_k^0; w_N\right)$. Therefore, the departure time from the queue (i.e., initiation of service) of the customer arriving at T_k^0 in the vacation system occurs no earlier than the departure time from the queue of the customer arriving at time T_k^0 in the *GI/GI/c* queue. Consequently, we conclude that the set of customers waiting in the queue in the *GI/GI/c* system at time t is a subset of the set of customers waiting in the queue in the vacation system at the same time. Similarly, we consider $Q_u(t-u; z^-) \leq Q_u(t-u; z^+)$, which is easier to establish, since, for $k \geq N$ (with the earlier definition of T_N^0 and w_N),

$$W_N\left(T_k^0; w_N\right) \geq W_N\left(T_k^0; 0\right).$$

So the set of customers waiting in the queue in the lower bound *GI/GI/c* system at time t is a subset of the set of customers waiting in the queue in the upper bound *GI/GI/c* system at the same time. □

Appendix: A list of selected notation

Notation	Meaning
$A_n : n \in \mathbb{Z}\backslash\{0\}$	Interarrival time between the arrivals at T_n^0 and $T_n^{0,+}$ (Sect. 2.1)
$D_n^0 : n \in \mathbb{Z}\backslash\{0\}$	Waiting time of the customer arriving at T_n^0 in the vacation system (Sect. 2.1.2)
$E_u(t; z) : t \geq 0$	Time elapsed since previous arrival at time $u + t$ when the time elapsed since previous arrival at time u is the corresponding subvector of z, i.e., e (Sect. 2.2.1)
$i(n) : n \in \mathbb{Z}\backslash\{0\}$	Label of the server who serves the customer arriving at T_n^0 (Sect. 2.1.2)
$M(t) : t \geq 0$	Running time maximum of process $\{X(s) : s \geq t\}$ (Sect. 4)
$N_u^0(t) : t \geq 0$	Number of arrivals during $[u, u + t]$ (Sect. 2.1)
$N^0(t)$	Number of arrivals during $[0, t]$ if $t \geq 0$ or in $[t, 0]$ if $t < 0$ (Sect. 2.1)
$N_u^i(t) : t \geq 0$	Number of activities initiated by server i during $[u, u + t]$ (Sect. 2.1)
$N^i(t)$	Number of activities initiated by server i during $[0, t]$ if $t \geq 0$ or $[t, 0]$ if $t < 0$ (Sect. 2.1)
$Q_u(t; z) : t \geq 0$	Number of people waiting in queue at time $u + t$ of a *GI/GI/c* queue that starts at time u and is initialized with z (Sect. 2.2.1)
$Q_v(t)$	Number of people waiting in queue in stationary vacation system at time t (Sect. 2.1.1)
$R_u(t; z) : t \geq 0$	Ordered (non-decreasing) remaining service times of all c servers at time $u + t$ of a *GI/GI/c* queue that starts at time u and is initialized with z (Sect. 2.2.1)
$\{S_n^{(0)} : n \geq 0\}$	Random walk with negative drift associated with the arrival renewal process (Sect. 4)
$\{S_n^{(i)} : n \geq 0\}$	Random walk with negative drift associated with the activity renewal process of server i (Sect. 4)
$T_n^0 : n \in \mathbb{Z}\backslash\{0\}$	n-th ($(-n)$-th) arrival time counting forward (backward) in time from time 0 (Sect. 2.1)
$T_n^{0,+} : n \in \mathbb{Z}\backslash\{0\}$	Next arrival time after T_n^0 (Sect. 2.1)
$T_n^{0,-} : n \in \mathbb{Z}\backslash\{0\}$	Previous arrival time before T_n^0 (Sect. 2.1)
$T_n^i : n \in \mathbb{Z}\backslash\{0\}$	n-th ($(-n)$-th) activity initiation time of server i counting forward (backward) in time from time 0 (Sect. 2.1)
$T_n^{i,+} : n \in \mathbb{Z}\backslash\{0\}$	Next activity initiation time of server i after T_n^i (Sect. 2.1)
$T_n^{i,-} : n \geq \mathbb{Z}\backslash\{0\}$	Previous activity initiation time of server i before T_n^i (Sect. 2.1)
$U^0(t)$	Time until next arrival from time t (Sect. 2.2.2)
$U^i(t)$	Time until next activity initiated by server i from time t (Sect. 2.2.2)
$U(t)$	c-dimensional vector $\left(U^1(t), \ldots, U^c(t)\right)^T$ (Sect. 2.2.2)
$V_n : n \in \mathbb{Z}\backslash\{0\}$	Service time of the customer who arrives at T_n^0 (Sect. 2.1.1)
$V_n^i : n \in \mathbb{Z}\backslash\{0\}$	Length of the activity of server i that is initiated at T_n^i (Sect. 2.1)
$W(n) : n \in \mathbb{Z}\backslash\{0\}$	Kiefer–Wolfowitz workload vector at time T_n^0 of a stationary *GI/GI/c* queue (Sect. 2.2.1)
$W_k\left(T_n^0; w\right)$: $n \geq k$ and $n, k \in \mathbb{Z}\backslash\{0\}$	Kiefer–Wolfowitz workload vector at time T_n^0 of a *GI/GI/c* queue which has its workload vector at time T_k^0 being w (Sect. 2.2.1)
$W_v(n) : n \in \mathbb{Z}\backslash\{0\}$	Analog Kiefer–Wolfowitz workload vector at time T_n^0 of the stationary vacation system (Sect. 2.2.2)
$W_v(n+) : n \in \mathbb{Z}\backslash\{0\}$	Analog Kiefer–Wolfowitz workload vector at time $T_n^{0,+}$ of the stationary vacation system (Sect. 2.2.2)
$X(t)$	$X_0(t)$ if $t \geq 0$ or $X_t(-t)$ if $t < 0$ (Sect. 2.1.1)

Notation	Meaning
$X_u(t) : t \geq 0$	$N_u^0(t) - \sum_{i=1}^c N_u^i(t)$ (Sect. 2.1.1)
$Z(t)$	State vector of a stationary *GI/GI/c* queue at time t
$Z_u(t; z) : t \geq 0$	State vector at time $u + t$ of a *GI/GI/c* queue that starts at time u and is initialized with z (Sect. 2.2.1)
$\sigma_u^i(t) : t \geq 0$	Number of service initiations by server i during $[u, u + t]$ (Sect. 2.1.2)
$\Phi_j^i : j \geq 0, 0 \leq i \leq c$	j-th downward "milestone" of random walk $\{S_n^{(i)} : n \geq 0\}$ (Sect. 4)
$\Upsilon_j^i : j \geq 0, 0 \leq i \leq c$	j-th upward "milestone" of random walk $\{S_n^{(i)} : n \geq 0\}$ (Sect. 4)

References

1. Asmussen, S.: Applied Probability and Queues, 2nd edn. Springer, Berlin (2003)
2. Asmussen, S., Glynn, P., Thorisson, H.: Stationarity detection in the initial transient problem. ACM Trans. Model. Comput. Simul. (TOMACS) **2**(2), 130–157 (1992)
3. Blanchet, J., Chen, X.: Steady-state simulation of reflected Brownian motion and related stochastic networks (2013). arXiv preprint arXiv:1202.2062
4. Blanchet, J., Dong, J.: Perfect sampling for infinite server and loss systems. Adv. Appl. Probab. **47**(3), 761–786 (2014). Forthcoming
5. Blanchet, J., Sigman, K.: On exact sampling of stochastic perpetuities. J. Appl. Probab. **48**(A), 165–182 (2011)
6. Blanchet, J., Wallwater, A.: Exact sampling for the stationary and time-reversed queues. ACM Trans. Model. Comput. Simul. (TOMACS) **25**(4), 26:1–26:27 (2015)
7. Chen, H., Yao, D.: Fundamentals of Queueing Networks: Performance, Asymptotics and Optimization, vol. 46. Springer, Berlin (2013)
8. Connor, S., Kendall, W.: Perfect simulation for a class of positive recurrent Markov chains. Ann. Appl. Probab. **17**(3), 781–808 (2007)
9. Connor, S., Kendall, W.: Perfect simulation of M/G/c queues. Adv. Appl. Probab. **47**(4), 1039–1063 (2015)
10. Corcoran, J., Tweedie, R.: Perfect sampling of ergodic Harris chains. Ann. Appl. Probab. **11**(2), 438–451 (2001)
11. Ensor, K., Glynn, P.: Simulating the maximum of a random walk. J. Stat. Plan. Inference **85**, 127–135 (2000)
12. Foss, S.: On the approximation of multichannel service systems. Sibirsk. Mat. Zh. **21**(6), 132–140 (1980)
13. Foss, S., Chernova, N.: On optimality of the FCFS discipline in multiserver queueing systems and networks. Sib. Math. J. **42**(2), 372–385 (2001)
14. Foss, S., Konstantopoulos, T.: Lyapunov function methods. Lecture Notes. http://www2.math.uu.se/~takis/L/StabLDC06/notes/SS_LYAPUNOV.pdf (2006)
15. Foss, S., Tweedie, R.: Perfect simulation and backward coupling. Stoch. Models **14**, 187–203 (1998)
16. Garmarnik, D., Goldberg, D.: Steady-state GI/GI/n queue in the Halfin–Whitt regime. Ann. Appl. Probab. **23**, 2382–2419 (2013)
17. Hillier, F.S., Lo, F.D.: Tables for multiple-server queueing systems involving Erlang distributions. Tech. Rep. 31, Department of Operations Research, Stanford University (1971)
18. Kelly, F.: Reversibility and Stochastic Networks, vol. 40. Wiley, Chichester (1979)
19. Kendall, W.: Perfect simulation for the area-interaction point process. In: Accardi, L., Heyde, C.C. (eds.) Probability towards 2000, pp. 218–234. Springer, New York (1998)
20. Kendall, W.: Geometric ergodicity and perfect simulation. Electron. Comm. Probab. **9**, 140–151 (2004)
21. Kendall, W., Møller, J.: Perfect simulation using dominating processes on ordered spaces, with application to locally stable point processes. Adv. Appl. Probab. **32**(3), 844–865 (2000)
22. Liu, Z., Nain, P., Towsley, D.: Sample path methods in the control of queues. Queueing Syst. **21**(1–2), 293–335 (1995)

23. Propp, J., Wilson, D.: Exact sampling with coupled Markov chains and applications to statistical mechanics. Random Struct. Alg. **9**, 223–252 (1996)
24. Rubinstein, R., Kroese, D.: Simulation and the Monte Carlo method, vol. 707. Wiley, New York (2011)
25. Sigman, K.: Exact simulation of the stationary distribution of the FIFO M/G/c queue. J. Appl. Probab. **48A**, 209–216 (2011)
26. Sigman, K.: Exact sampling of the stationary distribution of the FIFO M/G/c queue: the general case for $\rho < c$. Queueing Syst. **70**, 37–43 (2012)
27. Wolff, R.: An upper bound for multi-channel queues. J. Appl. Probab. **14**, 884–888 (1977)